z/VM
7.3

CMS Commands and Utilities Reference

SC24-6260-73

Note:

Before you use this information and the product it supports, read the information in “Notices” on page 1421.

This edition applies to version 7, release 3 of IBM® z/VM® (product number 5741-A09) and to all subsequent releases and modifications until otherwise indicated in new editions.

Last updated: 2023-05-16

Contents

Figures

Tables

About This Document

This document provides reference information about CMS commands and utilities for IBM® z/VM®. It provides the format, operand and option descriptions, and usage information for each general-use CMS command and for the format words of the CMS HELP Facility.

Some CMS commands are documented in other documents. See "CMS Commands Not Described in This Document" on page 20.

Intended Audience

This document is for anyone who uses CMS commands. It is particularly intended for those who need comprehensive, detailed descriptive information about the commands.

This document is primarily intended as a source of reference information for those who already have some knowledge of how to use CMS commands. For this document to be the most useful, you should already be familiar with *z/VM: CMS User's Guide*. The *z/VM: CMS User's Guide* contains extended information about how to work in a CMS environment and how to use commands to perform different tasks. For many of the commands described in this document, you may find additional useful notes in *z/VM: CMS User's Guide*.

If you are using CMS for the first time, it would be helpful to see *z/VM: CMS Primer* for introductory tutorial information about CMS.

Where to Find More Information

You can find more information about CMS in the documents listed in the "Bibliography" on page 1425.

Links to Other Documents and Websites

The PDF version of this document contains links to other documents and websites. A link from this document to another document works only when both documents are in the same directory or database, and a link to a website works only if you have access to the Internet. A document link is to a specific edition. If a new edition of a linked document has been published since the publication of this document, the linked document might not be the latest edition.

How to provide feedback to IBM

We welcome any feedback that you have, including comments on the clarity, accuracy, or completeness of the information. See How to send feedback to IBM for additional information.

Summary of Changes for z/VM: CMS Commands and Utilities Reference

This information includes terminology, maintenance, and editorial changes. Technical changes or additions to the text and illustrations for the current edition are indicated by a vertical line (**|**) to the left of the change.

SC24-6260-73, z/VM 7.3 (May 2023)

This edition includes changes to support product changes provided or announced after the general availability of z/VM 7.3.

[VM66453, VM66457, PH51239] CMS Password/Key Management Utility - Keyvault

With the PTFs for APARs VM66453 (CMS), VM66457 (VMSES/E), and PH51239 (TCP/IP), z/VM 7.3 provides support for a CMS password/key management utility called KEYVAULT, which allows applications to securely store and retrieve user ID keys (logon passwords). z/VM Centralized Service Management (z/VM CSM) and the TCP/IP FTP client are updated to use the new KEYVAULT utility for automated remote host login procedures.

The following CMS utility is new:

- "KEYVAULT" on page 1331

[VM66646] Security Settings and Compliance Interfaces

With the PTF for APAR VM66646, z/VM 7.3 provides security settings and compliance API and command interfaces for compliance status extractors. The output from these new extractor interfaces contains security-relevant configuration data that can be analyzed for Payment Card Industry Data Security Standard (PCI DSS) compliance or for adherence to security configuration baselines. The API is provided through a new Systems Management interface that passes data to the extractor program.

The following CMS utility is new:

- "COMPEXTR" on page 1310

SC24-6260-73, z/VM 7.3 (September 2022)

This edition supports the general availability of z/VM 7.3. Note that the publication number suffix (-73) indicates the z/VM release to which this edition applies.

Removal of the CMSDESK function, external GUI functions, and the GUICSLIB DCSS

The CMSDESK function, external GUI functions, and the GUICSLIB DCSS are removed. The CMS CMSDESK command, the CMS SET WORKSTATION command, and the CMS QUERY WORKSTATION command are no longer valid commands. References to CMS GUI are removed from the publications.

Documentation for the following commands is updated:

- "QUERY" on page 620
- "QUERY SYSNAMES" on page 764
- "SET" on page 903
- "SET SYSNAME" on page 992
- "XEDIT" on page 1258

The following commands are deleted:

- CMSDESK
- QUERY WORKSTATION
- SET WORKSTATION

SC24-6260-03, z/VM 7.2 (May 2022)

This edition includes changes to support product changes provided or announced after the general availability of z/VM 7.2.

Miscellaneous updates for May 2022

References to IBM Z® Application Assist Processor (zAAP), which is not supported on IBM z13 and later models, are removed. The following topic is updated:

- "ACCOUNT" on page 1292

SC24-6260-03, z/VM 7.2 (December 2021)

This edition includes changes to support product changes provided or announced after the general availability of z/VM 7.2.

[PH40080, VM66561, VM66581] Query SSL GSKKYMAN Certificates

With the PTFs for APARs PH40080 (TCP/IP), VM66561 (CMS), and VM66581 (VMSES/E), z/VM provides support in TCP/IP for querying certificates within a specific GSKKYMAN certificate database. The query lists certificate labels and displays certain attributes of the certificates.

The following section is updated:

- "DEFAULTS" on page 153

Miscellaneous updates for December 2021

The example of the RESERVE command is updated. See "RESERVE" on page 829.

SC24-6260-02, z/VM 7.2 (September 2021)

This edition includes terminology, maintenance, and editorial changes.

Miscellaneous updates for September 2021

Messages DMS1498E and DMS1499E are added to the VMFPLCD command. See "VMFPLCD" on page 1123.

SC24-6260-02, z/VM 7.2 (July 2021)

This edition includes changes to support product changes provided or announced after the general availability of z/VM 7.2.

[VM66511, VM66512, VM66513] Help File Improvements

With the PTFs for APARs VM66511 (CP), VM66512 (CMS), and VM66513 (VMSES), the z/VM Help facility adds information for CP Directory Statements and for CP System Configuration Statements. HELP files of type HELPDIRE are added to support the new DIRECTORY HELP component. HELP files of type HELPSYSC are added to support the new SYSCONFIG HELP component. For a list of z/VM HELP components and the associated HELP file types and high-level HELP menus, see Table 19 on page 385.

Miscellaneous updates for July 2021

Information about the HELP command is enhanced and clarified. See “HELP” on page 381.

SC24-6260-01, z/VM 7.2 (September 2020)

This edition includes changes to support the general availability of z/VM 7.2.

Updates reflect the removal of KANJI language files from base z/VM components. The only currently supported languages are American English and uppercase English.

Chapter 1. Introduction and General Concepts

z/VM has two main command languages, which correspond to these components of the z/VM system:

Control Program (CP)
controls the resources of the real machine; that is, the physical computer system. For more information on the CP commands, see *z/VM: CP Commands and Utilities Reference*.

Conversational Monitor System (CMS)
is an interactive user interface that runs under CP. CMS can simulate many of the functions of OS (Operating System) and DOS (Disk Operating System). This lets you run many OS and DOS programs in a conversational environment. This book describes general-use CMS commands you can use in the CMS environment. 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.

CMS Environment

The CMS command language lets you create, modify, and, in general, handle a system of files.

The OS/VS Assembler and many OS/VS and VSE (DOS) language processors can run 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 run under CMS. When you enter the appropriate CMS commands, CMS invokes the assembler and the compilers. This book describes the ASSEMBLE command, and the supported compiler commands are described in the appropriate Licensed Program books.

CMS commands let you read cards from a virtual card reader, punch cards to a virtual card punch, and print records on a virtual printer. There are also many commands to help you handle your virtual disks, your directories in file pools, and the files on your minidisks and directories.

A special set of CMS commands is available to you when you enter:

```
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.

The HELP format words create HELP ‘text’ information for user-defined commands, execs, and messages. For more information on the function, formats, and operands of the z/VM HELP Facility format words, see Chapter 5, “HELP and HELPCONV Format Words,” on page 1397.

In addition to the messages listed with each command, many CMS commands can issue system messages. For more information on these messages, see Chapter 6, “System Messages,” on page 1411.

Syntax, Message, and Response Conventions

The following topics provide information on the conventions used in syntax diagrams and in examples of messages and responses.

How to Read Syntax Diagrams

Special diagrams (often called *railroad tracks*) are used to show the syntax of external interfaces.

To read a syntax diagram, follow the path of the line. Read from left to right and top to bottom.

- The ►►—— symbol indicates the beginning of the syntax diagram.

- The ——► symbol, at the end of a line, indicates that the syntax diagram is continued on the next line.
- The ►—— symbol, at the beginning of a line, indicates that the syntax diagram is continued from the previous line.
- The ——►◄ symbol indicates the end of the syntax diagram.

Within the syntax diagram, items on the line are required, items below the line are optional, and items above the line are defaults. See the examples in Table 1 on page 2.

Table 1. Examples of Syntax Diagram Conventions

Syntax Diagram Convention	Example
Keywords and Constants A keyword or constant appears in uppercase letters. In this example, you must specify the item KEYWORD as shown. In most cases, you can specify a keyword or constant in uppercase letters, lowercase letters, or any combination. However, some applications may have additional conventions for using all-uppercase or all-lowercase.	►► KEYWORD ►◄
Abbreviations Uppercase letters denote the shortest acceptable abbreviation of an item, and lowercase letters denote the part that can be omitted. If an item appears entirely in uppercase letters, it cannot be abbreviated. In this example, you can specify KEYWO, KEYWOR, or KEYWORD.	►► KEYWOrd ►◄
Symbols You must specify these symbols exactly as they appear in the syntax diagram.	***** Asterisk **:** Colon **,** Comma **=** Equal Sign **-** Hyphen **()** Parentheses **.** Period
Variables A variable appears in highlighted lowercase, usually italics. In this example, *var_name* represents a variable that you must specify following KEYWORD.	►► KEYWOrd — *var_name* ►◄

Table 1. Examples of Syntax Diagram Conventions (continued)

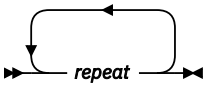
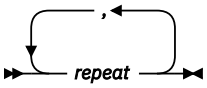
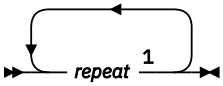
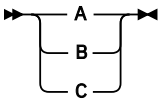
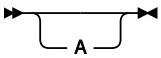
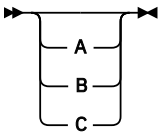
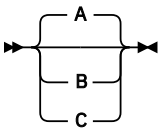
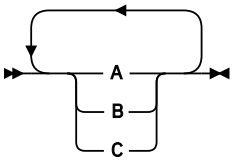
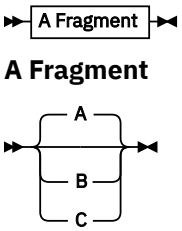
Syntax Diagram Convention	Example
Repetitions An arrow returning to the left means that the item can be repeated. A character within the arrow means that you must separate each repetition of the item with that character. A number (1) by the arrow references a syntax note at the bottom of the diagram. The syntax note tells you how many times the item can be repeated. Syntax notes may also be used to explain other special aspects of the syntax.	   Notes: [1] Specify *repeat* up to 5 times.
Required Item or Choice When an item is on the line, it is required. In this example, you must specify A. When two or more items are in a stack and one of them is on the line, you must specify one item. In this example, you must choose A, B, or C.	 
Optional Item or Choice When an item is below the line, it is optional. In this example, you can choose A or nothing at all. When two or more items are in a stack below the line, all of them are optional. In this example, you can choose A, B, C, or nothing at all.	 
Defaults When an item is above the line, it is the default. The system will use the default unless you override it. You can override the default by specifying an option from the stack below the line. In this example, A is the default. You can override A by choosing B or C.	
Repeatable Choice A stack of items followed by an arrow returning to the left means that you can select more than one item or, in some cases, repeat a single item. In this example, you can choose any combination of A, B, or C.	

Table 1. Examples of Syntax Diagram Conventions (continued)	
Syntax Diagram Convention	**Example**
Syntax Fragment Some diagrams, because of their length, must fragment the syntax. The fragment name appears between vertical bars in the diagram. The expanded fragment appears in the diagram after a heading with the same fragment name. In this example, the fragment is named "A Fragment."	

Examples of Messages and Responses

Although most examples of messages and responses are shown exactly as they would appear, some content might depend on the specific situation. The following notation is used to show variable, optional, or alternative content:

xxx
Highlighted text (usually italics) indicates a variable that represents the data that will be displayed.

[ ]
Brackets enclose optional text that might be displayed.

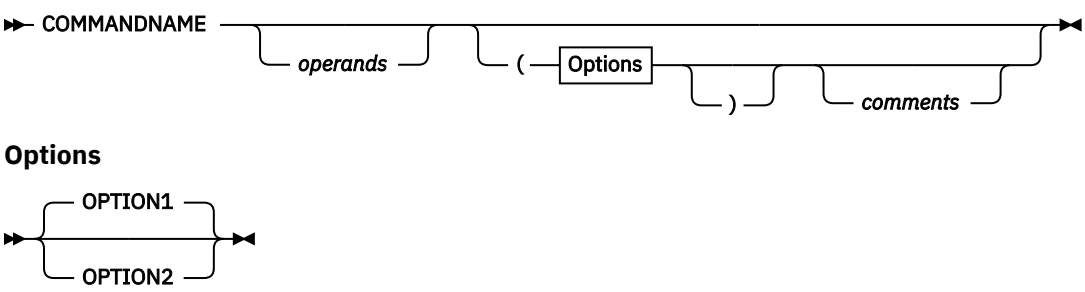
{ }
Braces enclose alternative versions of text, one of which will be displayed.

|
The vertical bar separates items within brackets or braces.

...
The ellipsis indicates that the preceding item might be repeated. A vertical ellipsis indicates that the preceding line, or a variation of that line, might be repeated.

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 options list. The general format of CMS commands is:



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 “Syntax, Message, and Response Conventions” on page 1.

Command Name

The command name is an alphanumeric symbol of one-to-eight characters. In general, a command name is based on a verb or verb-noun combination that describes the function the system does. For example, you may want to find out information concerning your files. In this case, you would use either the FILELIST or the LISTFILE command. If a command name is eight characters long, any character entered

in the command name field after the eighth character is ignored. Anything entered after a blank following the command name (or the second word of a two-word command name) is treated as a command operand.

Two-word Commands

Some CMS commands have two-word names, for example, CREATE DIRECTORY, DELETE LOCK, and so forth. These require special attention if you have created user-written commands (modules) or user-written nucleus extensions. These two-word commands may use the same first word as your user-written commands. If you have written nucleus extensions, you must be aware that each two-word command maps to a one-word module name. Your nucleus extension name should be the same as the module name, not the command name.

Command Operands

The command operands are keywords and positional operands of one to eight alphanumeric characters each. One notable exception is when the operand is a directory identifier (*dirid*). A *dirid* may be up to 153 characters. In certain other 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 does 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, you may use blanks to separate the last operand from the options list. CMS recognizes a left parenthesis "(" as the beginning of an options list; it does not have to start with a blank.

Command Options

Command options are keywords that control the execution of the command. The command formats in this book show all the options for each CMS command.

A left parenthesis begins the options list. If you specify options, the parenthesis is required. It does not need to be followed by a blank. An optional right parenthesis ends the options list; it does not need to be preceded by a blank. CMS always treats left and right parentheses as separate tokens. The majority of CMS commands ignore any text following the right parenthesis, so comments may be entered there in these cases.

Note: CMS commands entered from the command line are generally limited to 64 eight-character tokens. This is known as the tokenized PLIST. However, not all commands use the tokenized PLIST. Those that do not are limited to 255 characters in length. This is known as the extended PLIST. You would seldom encounter this limit because the size of the command line itself is limited.

On commands issued from a program, there is generally no limit, although some commands do enforce the 64 token limit. Still other special commands, like TELL and XEDIT, are not subject to either of these rules.

Character Set Usage

You can enter CMS commands using a combination of characters from six different character sets. Table 2 on page 5 shows the contents of each character set.

Table 2. Character Sets and Their Contents

Character Set	Names	Symbols
Separator	Blank	
National	Dollar Sign Pound Sign At Sign	$ # @

Table 2. Character Sets and Their Contents (continued)

Character Set	Names	Symbols
Alphabetic	Uppercase Lowercase	A through Z a through z
Numeric	Numeric	0 through 9
Alphanumeric	National Alphabetic Numeric	$, #, @ A through Z a through z 0 through 9
Special		All other characters

When to Enter CMS Commands

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 by means of an attention interruption. The system acknowledges the interruption by unlocking the keyboard. Now you can enter commands.

If your terminal is a display device, you will have no problem entering commands while the virtual machine is busy because its keyboard remains unlocked for additional command input.

Note: In these circumstances the command you enter is stacked in the terminal input buffer and is not executed until the command currently being executed completes. If more commands are entered than CP can handle, a NOT ACCEPTED message is displayed at the display terminal.

Naming CMS Files

When you create a file, you can give it any file name and file type you choose, subject to the following rules:

- 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: The default line end character is a #, so if you want to use a # in a name, you may have to change your linend character. Use the CP commands QUERY TERM and TERM LINEND to find out what your current linend character is and to change it if necessary.

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 name it whatever you choose, subject to the following rules:

- The complete directory name (also referred to as *dirname*) is made up of its parent directory's name and any subdirectory names, with each name separated by a period.
- A directory name can be from one-to-sixteen characters.
- The valid characters are A-Z, a-z, 0-9, $, #, @, _ (underscore).
- Two or more subdirectories may have the same name if each has a different parent directory.

This example illustrates the preceding rules:

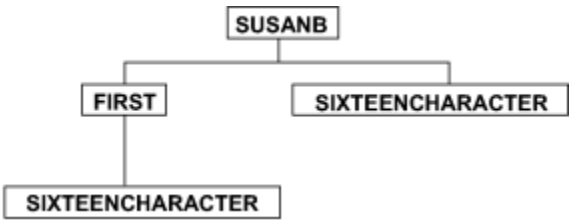


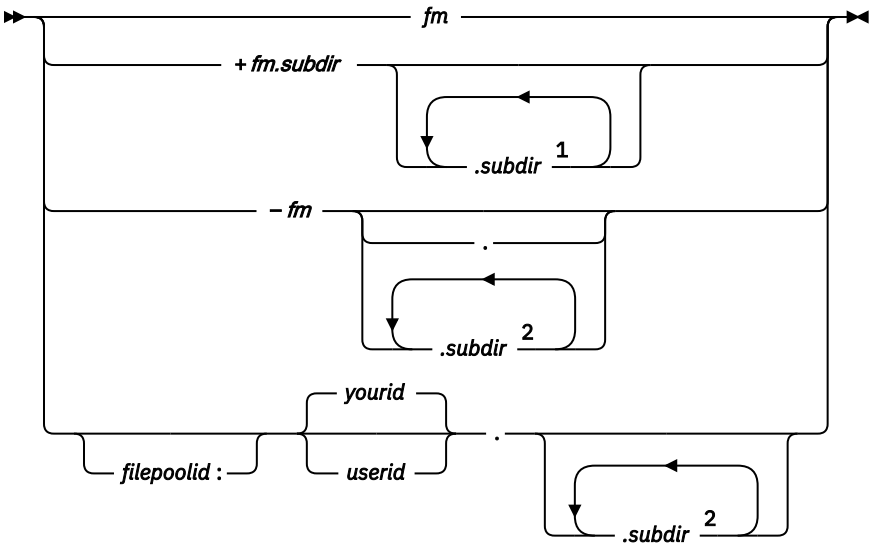
Figure 1. Example of a Directory Structure

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. Because it would be time-consuming to enter the complete directory name for every *dirid* in a command format, a shorthand notation is allowed. There are four ways you can represent a *dirid*:

Ways to Specify a Dirid



Notes:

[1] A total of 7 can be specified.
[2] A total of 8 can be specified.

Where:

fm
is the file mode letter assigned to the directory when it is accessed with the ACCESS command.

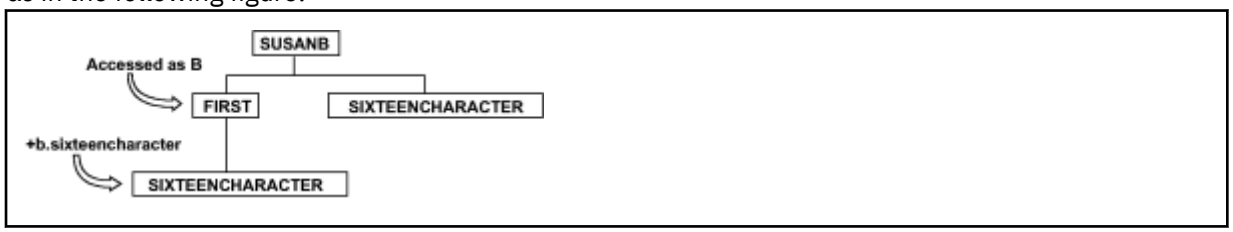
Note: Do not use a file mode number to specify *fm* unless you are specifying a file or set of files. (File mode numbers are attributes of files, not SFS directories or minidisks.) You cannot specify file mode numbers on commands that operate on an entire minidisk or SFS directory. For example, do not use file mode numbers on such commands as ACCESS, DIRLIST, FORMAT, RELOCATE, and RELEASE unless you are specifying a set of files.

+fm.subdir...
represents the name of a directory. The *+fm* means to start at the directory accessed as *fm* and go down the directory structure. You can specify up to 7 additional subdirectories, and together with the top directory 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 *subdir* 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
```

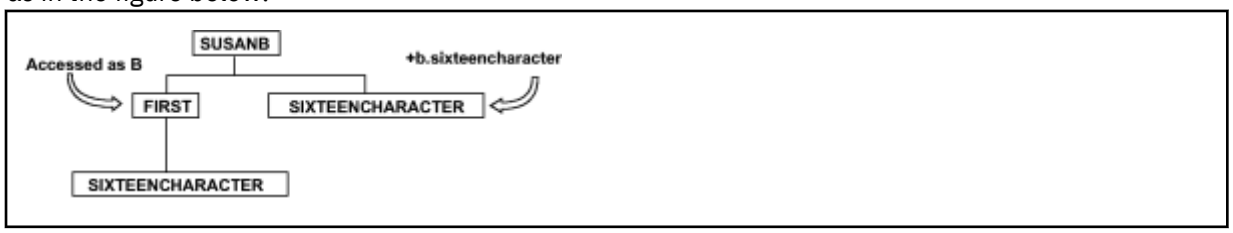
as in the following figure:



–fm...
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. You can specify up to 8 lower level directories (subdirectories), as described above. An example of this type of *dirid* is:

```
–b.sixteencharacter
```

as in the figure below:



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

filepoolid
is the name of the file pool the directory resides in. If not specified, the default file pool ID you set with the SET FILEPOOL command is used (the system does not supply a default). You (or the system administrator) can also set a default file pool in your user CP directory. This way you do not have to enter the SET FILEPOOL command each time you log on. The *filepoolid* can be up to eight characters long. The first character must be alphabetic, but the remaining characters can be alphabetic or numeric. You can enter both upper- and lowercase characters, but all characters are treated as uppercase. The *filepoolid* must end with a colon if it is being used as part of a directory name. Otherwise, it need not end with a colon.

userid
is both the user ID for the owner of the directory and the name of the user's top directory. This name is also referred to as the file space or file space ID. The user ID can be up to 8 characters long. Lowercase is converted to uppercase. If it is not specified, either the virtual machine user ID of the issuer of the routine is filled in or it is taken from the input to the SET FILESPACE command.

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

```
access . a
```

accesses your top directory as A.

subdir1.subdir2...subdir7
are the directory levels. *userid* is the top directory and *subdir1* through *subdir7* are subdirectories. The valid characters in a subdirectory name are A-Z, a-z, 0-9, $, _, #, @. Lower case is converted to upper case.

Note: The default line end character is a #, so if you want to use a # in a name, you may have to change your linend character. Use the CP commands QUERY TERM and TERM LINEND to find out what your current linend character is and to change it if necessary.

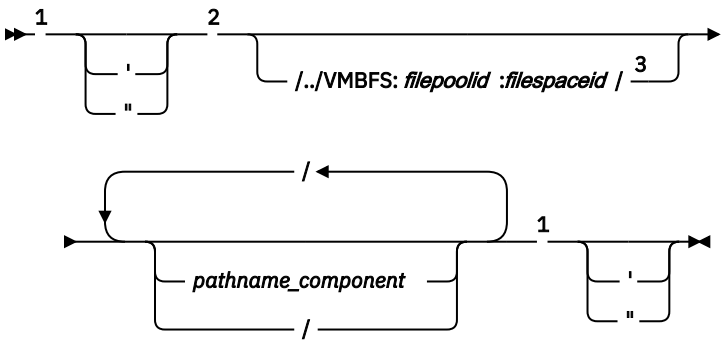
Understanding Byte File System (BFS) Path Name Syntax

All objects (files, directories, and so on) in the OpenExtensions byte file system (BFS) are identified through path names. A path name identifies the object within the BFS hierarchy by specifying the directories leading to the object.

A BFS path name can represent a file system accessed through the Network File System (NFS). The NFS file system can be on a remote or local system, which can be VM or non-VM. The OPENVM MOUNT command links an NFS file system to a BFS path name, enabling it to be used on most commands and interfaces that accept BFS path names.

For simplicity in command syntax, the BFS path name identifier is usually shown as the variable, *pathname*.

Format



Notes:

[1] A single quotation mark or a double quotation mark is an optional delimiter. If specified to signify the start of a path name, the identical delimiter also must be specified to signify the end of the path name. (See usage note “7” on page 12.)

[2] The minimum path name is a single slash (/).

[3] The ending slash is required only if one or more path name components are also specified.

Operands

/../VMBFS:*filepoolid*:*filespaceid*/
is a construct that identifies the byte file system. It is referred to as the fully qualified BFS root.

/../VMBFS:
is a keyword string that indicates this is an OpenExtensions byte file system. This string is not case sensitive and must end with a colon (:).

***filepoolid*:**
is the name of the file pool that contains the BFS data. The file pool name can be up to eight characters long and is not case sensitive. The first character must be alphabetic, but the remaining characters can be alphabetic or numeric. The name must be followed by a colon (:).

***filespaceid*/**
is the name of the file space where the BFS resides. The file space ID can be up to eight characters long and is not case sensitive. The name must be followed by a slash (/) if one or more path name components are also specified.

pathname_component
is the name of an object in the BFS hierarchy. Each path name component can be 1-255 characters in length. The slash character (/) and the null character (X'00') are not valid within a path name component. Path name component names are case sensitive.

When multiple path name components are specified, they must be separated by slashes.

All path name components prior to the last one specified will be interpreted as directory names in the hierarchy. The last path name component, when not followed by a slash, can be a directory or another type of object. If the last path name component is followed by a slash, it will always be interpreted as a directory.

/
when specified as a single character path name, indicates the root (top) directory of the currently mounted byte file system. In the OpenExtensions environment, the root directory can be assigned by using the OPENVM MOUNT command, or by the POSIXINFO FSROOT statement in your user directory entry.

//
when a path name starts with exactly two slashes, it is not considered to be a BFS name. This type of path name is interpreted as a CMS record file system name by the functions that support redirection to the CMS record file system. When such a name is given as a parameter to a command or function that does not support redirection to the CMS record file system, the request will be rejected.

For example, the shell command:

```
$ c89 pgm.c -o //mymod.module.a
```

will create the file MYMOD MODULE A on your A-disk. The OPENVM command:

```
openvm get ./test/book/ch1.scr   //chapter1.script.a
```

will fail with an error message indicating the file name is not valid.

Note: A path name must not start with two slashes when in the XEDIT environment.

Usage Notes

1. A byte file system can be enrolled in the same file pool as other byte file systems and SFS users.
2. In the OpenExtensions environment, all byte file systems are uniquely identified with the /../vmbfs:*filepoolid*:*filespaceid* construct.
3. Path names can be specified in several ways:
   - When the first character of the path name is not a slash, the path name is known as a *relative path name*. The search for the BFS object starts at the working directory. To establish the working directory, use the OPENVM SET DIRECTORY command. To find the value of the current working directory, use the OPENVM QUERY DIRECTORY command.
   - When /../vmbfs:*filepoolid*:*filespaceid*/ is specified at the start of a path name, it is referred to as a *fully qualified path name*. The object is searched for in the byte file system, which is defined as file space *filespaceid* in file pool *filepoolid*. The byte file system does not need to be explicitly mounted.
   - If the path name starts with a slash (but not /../vmbfs:*filepoolid*:*filespaceid*/), the path name is known as an *absolute path name*. The search for the object starts from the root of the currently mounted byte file system. The root directory can be established by using the OPENVM MOUNT command, or by the POSIXINFO FSROOT statement in your user directory entry. To find the value of the root directory, use the OPENVM QUERY MOUNT command.

   For more information on OPENVM commands, see *z/VM: OpenExtensions Commands Reference*. For more information on user directory statements, see *z/VM: CP Planning and Administration*.
4. The entire path name must be in the range of 1-1023 characters. Individual path name components cannot exceed 255 characters. All characters are valid within a path name, with the following restrictions:

- The null character (X'00') is not permitted within a path name.
- A slash (/) is interpreted as the delineator of a path name component.

For an application to be portable to the broadest set of environments, POSIX standards suggest that the application restrict the maximum length of a BFS path name component to 14 characters and use only the following characters:

A-Z
Uppercase alphabetic

a-z
Lowercase alphabetic

0-9
Numeric

.
Period

_
Underscore

–
Dash

5. Path name components are case sensitive. For example, Abc, abC, and ABC are valid unique path name components. When a path name is entered on the CMS command line, it will not be uppercased. However, a path name entered on the XEDIT command line will be uppercased when SET CASE UPPER is in effect.
6. There are two BFS path name components that have special meaning during path name resolution. These are:

.
The path name component consisting of a single dot character (`.`) refers to the directory specified by the preceding path name component.

Some dot (.) examples:

a. If you specified a path name of:

```
'/joes/recipes/./pie'
```

It would be equivalent to:

```
'/joes/recipes/pie'
```

b. If you specified a path name of:

```
'./joes'
```

It would be equivalent to:

```
'joes'
```

..
The path name component consisting of two dot characters (`..`), known as dot-dot, refers to the parent directory of its predecessor. As a special case, in the root directory, dot-dot refers to the root directory itself. The construct `/../vmbfs:`*filepoolid*`:`*filespaceid*`/` is the only exception.

Some dot-dot (. .) examples:

a. If you had previously set your working directory (using OPENVM SET DIRECTORY) to:

```
/joes/recipes/
```

And you specified a relative path name of `../tools`, this would be equivalent to specifying an absolute path name of:

```
/joes/tools
```

b. If you are working in `/bin/util/src`, and you want to go to `/bin/util`, you can enter:

```
openvm set directory ..
```

c. If you are working in `/u/rexx/prog/src`, and you want to refer to the file `test` in the directory `/u/rexx/appl/examples`, you could use the following path name to refer to that file:

```
../../appl/examples/test
```

7. Enclose a BFS path name within single quotation marks ('*pathname*') or double quotation marks ("*pathname*") if it contains any of the following characters. Results are unpredictable if a path name or path name component contains any of these characters and it is not enclosed within quotation marks.

   A blank space

   (
   Left parenthesis

)
   Right parenthesis

   '
   A single quotation mark

   "
   A double quotation mark

   *****
   Asterisk

   =
   Equal sign

   Note:

   a. If a path name includes a single quotation mark, specify the path name in one of these ways:
      - Place double quotation marks around the path name.
      - Place single quotation marks around the path name, but be sure to use two additional single quotation marks to denote the single quotation mark that is part of the path name.

   b. If a path name includes a double quotation mark, specify the path name in one of these ways:
      - Place single quotation marks around the path name.
      - Place double quotation marks around the path name, but be sure to use two additional double quotation marks to denote the double quotation mark that is part of the path name.

   c. All characters are taken literally; no symbolic substitution is done.

   Some examples:

   a. To list files in a directory called `my  dir` that is directly under your root directory, you must specify:

      ```
      openvm listfile '/my dir'
      ```

      Note that:

      ```
      openvm listfile /a/b/c
      ```

      is equivalent to:

```
openvm listfile '/a/b/c'
```

b. To list the files in a directory called `/a/b b'/c`, you can enter the name in either of the following ways:

```
openvm listfile '/a/b b''/c'
openvm listfile "/a/b b'/c"
```

c. To list the files in a directory called `/a/b b"/c`, you can enter the name in either of the following ways:

```
openvm listfile "/a/b b""/c"
openvm listfile '/a/b b"/c'
```

8. The OPENVM commands can be entered on a single line or on multiple lines. To enter multiple lines, type OPENVM and press the Enter key. You will get a message prompting you to enter more input lines. You must enter a null line to indicate the end of your command input. This is particularly useful for entering long path names.

   Leading and trailing blanks entered on an input line are preserved when multiple lines are put together. A blank is needed after a keyword and its following operand.

   This is an example of entering multiple lines:

```
openvm
```

   (Press the Enter key)

```
DMSWOV2140R Enter operands: (enter a null line to
            indicate that you are finished)

listfile
```

   (where LISTFILE is followed by a blank, and you press the Enter key)

```
'/A
/B
```

   (where /B is followed by a blank)

```
/c' (header
```

   (and press the Enter key twice to enter a null line)

   This is equivalent to the one-line command:

```
openvm listfile '/A/B /c' (header
```

   Because the path name is `/A/B /c` contains a blank, it must be enclosed in quotation marks on input.

9. Multiple adjacent slashes (//) in a path name (except the special case when a path name starts with exactly two slashes) are interpreted as a single slash by OPENVM commands. However, these multiple slashes are included in the maximum path name length check.

10. **Attention:**

    - You might need to change your terminal settings in order to specify a path name that contains certain special characters. For example, you want to use the # character in a name, but the default line end symbol is #. So you might have to change your logical line end symbol using the CP TERMINAL LINEND command.

      Use the QUERY LINEND and SET LINEND commands to find out and define your current line end character for full-screen CMS.

- If you choose to enclose a path name containing blanks in double quotation marks ("), you might need to use the CP TERMINAL ESCAPE command to change your logical escape symbol, because the default value is a double quotation mark.

  Use the CP QUERY TERMINAL command to display the special characters that are in effect for your terminal.

Online HELP Facility

The z/VM HELP Facility provides information intended for both new and experienced users. You can access the online HELP information with menus or with the CMS HELP command. HELP information is available for:

- Tasks
- Commands and subcommands
- Messages
- Routines and callable routines
- CMS Pipelines built-in programs
- REXX/VM, EXEC2, and EXEC statements
- Assembler language macros

Applications that run on z/VM can also provide HELP files.

Using the Online HELP Facility

You can receive online information about the commands described in using the z/VM HELP Facility. For example, to display a menu of CMS commands, enter:

```
help cms menu
```

To display information about a specific CMS command (ACCESS in this example), enter:

```
help cms access
```

You can also display information about a message by entering one of the following commands:

```
help msgid or help msg msgid
```

For example, to display information about message DMS001E, you can enter one of the following commands:

```
help dms001e or help msg dms001e
```

For more information about using the HELP Facility, see *z/VM: CMS User's Guide*. To display the main HELP Task Menu, enter:

```
help
```

For more information, see "HELP" on page 381.

Using Pattern Matching to Specify Sets of Files

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

*****
: is a placeholder that represents any number of character(s). You can use as many asterisks as necessary *anywhere* in a file name or file type, but the total number of characters, including the asterisks, cannot exceed eight.

For example, if you enter:

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

you are requesting 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     1DOGFILE A2
```

%

is a place holding character that represents any single character. You can use as many percent symbols as necessary anywhere in a file name or file type, but the total number of characters, including the asterisks, cannot exceed eight. For example, if you enter:

```
filelist %%% stock
```

you are requesting a file list that contains 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 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. Minidisks are searched for an exec with that name. (To find a file in a directory, read authority is required on both the file and directory.) If an exec is found, its file mode is compared to the file mode of the CMS installation saved segment. If the file mode of the saved segment is equal to or higher (closer to A) than the file mode of the directory or minidisk, the exec in the saved segment is executed. Otherwise, the exec in directory or on the minidisk is executed. However, if the exec is in a directory and the file is locked, the execution will fail with an error message.
   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.
   c. To find a file in a directory, read authority is required on both the file and the directory. If the file is in a directory and the file is locked, the processing fails with an error message.
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.

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.

Note: CMS will not search for an exec with the specified command name if SET IMPEX is OFF. For more information, see "SET IMPEX" on page 950.

When CMS searches for a translation or synonym (as in Steps 2 and 4), 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 search fails to find a CMS command, it is passed to CP for execution unless SET IMPCP is OFF. For more information, see "SET IMPCP" on page 949.

CMS Command Execution Characteristics

Table 3 on page 17 is an alphabetic list of the CMS commands that require special consideration when called from a user program. For example, a program running in low free storage can cause part of its own code to be overwritten if it calls a nonrelocatable CMS command, which also runs in low free storage. To avoid conflicts with nonrelocatable CMS commands, you should ensure your user programs are relocatable.

Any commands in this book not listed in this table are either nucleus-resident or disk-resident and relocatable. They will not interfere with the execution of a user program.

In Table 3 on page 17, the following letters indicate where the command executes:

E
Indicates this command is an exec. It may execute one or more CMS commands that run in the user free storage or transient areas. The transient area is the storage area used for temporary storage of programs or routines.

N
Indicates the first time this command is invoked it executes in the transient area. The transient area is the storage area used for temporary storage of programs or routines. On subsequent invocations, the command executes as a nucleus extension.

T
Indicates this command executes in the transient area. The transient area is the storage area used for temporary storage of programs or routines.

U
Indicates this command executes in low free storage. All OS free storage pointers are reset.

R
Indicates this command is relocatable.

Table 3. CMS Command Execution Characteristics

Command	Code	Command	Code
ALIALIST	E	MACLIB	R
AMSERV	U	MACLIST	E
ASSEMBLE	U	MODMAP	T
ASSGN	T	MOREHELP	E
AUTHLIST	E	MOVEFILE	R
CATCHECK	U	NAMES	E
CMSBATCH	U	NETLCNVT	E
CMSSERV	E	NOTE	E
COMPARE	T	NUCXDROP	T
CSLGEN	E	NUCXMAP	T
CSLLIST	E	OPTION	T
CSLMAP	E	OSRUN	R
CSRBDICV	E	PEEK	E
CSRCMPEV	E	PIPE	N
DDR	U	PSERV	U
DEFAULTS	E	PUNCH	T
DIRLIST	E	RDR	T
DISCARD	E	RDRLIST	E
DOSLIB	U	READCARD	T
DOSLKED	U	RECEIVE	E
DOSPLI	E	RESERVE	T
DSERV	U	RSERV	U
EDIT	E	RUN	E
ESERV	E	SENDFILE	E
EXECDROP	T	SETPRT	T
EXECMAP	R	SORT	R
EXECUPDT	E	SSERV	U
FCOBOL	E	SVCTRACE	T
FILELIST	E	SYNONYM	T
FORMAT	R	TAPEMAC	U
GENCMD	E	TAPPDS	U
GENDIRT	E	TELL	E
GENMSG	R	TXTLIB	R
GLOBAL	T	TYPE	T
HELPCONV	T	UPDATE	R
IOCP	U	VMFDOS	U

Table 3. CMS Command Execution Characteristics (continued)

Command	Code	Command	Code
LABELDEF	T	VMFLKED	E
LANGGEN	E	VMFMAC	E
LANGMERG	E	VMFPLCD	E
LISTDS	U	VMFPLC2	E
LISTIO	T	VMFMERGE	E
LKED	U	VMFTXT	E
LOADLIB	R	VMLINK	E

Special Types of CMS Commands

Immediate Commands

There are 10 Immediate commands that are handled in a different manner from other CMS commands.

Immediate commands can 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:

- IMMCMD macro in an assembler language program

  For more information, see *z/VM: CMS Macros and Functions Reference*.
- IMMCMD command within an exec (CMS EXEC, EXEC 2, REXX)

  For more information, see "IMMCMD" on page 404.
- NUCXLOAD command with the IMMCMD option specified

  For more information, see "NUCXLOAD" on page 580.

For more information, see "IMMCMD" on page 404.

Window Border Commands

Window border commands are single-character commands you enter in the corners of window borders to control the movement and display of windows.

The window 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

For more information, see "Window Border Commands" on page 1251.

Special CMS Commands Used within Other Commands

There are certain special commands that can be used only within other commands:

AUTHLIST
Displays authority information. Can be issued only from the DIRLIST or FILELIST environments.

ALIALIST
Displays alias information. Can be issued only from the FILELIST environment.

DISCARD
Erases a file, SFS directory, or MACLIB member that is displayed from the DIRLIST, FILELIST, MACLIST, RDRLIST, or PEEK command environments.

EXECUTE
Issues commands (or execs) from the CSLLIST, CSLMAP, DIRLIST, FILELIST, MACLIST, RDRLIST, or VMLINK environments.

The command *environments* where these special commands work are:

Command
Displays a list of...

CSLLIST
Callable Services Library (CSL) routines

CSLMAP
currently loaded Callable Services Library (CSL) routines

DIRLIST
Shared File System (SFS) directories

FILELIST
files and SFS directories

MACLIST
MACLIB members

RDRLIST
virtual reader files

For more information on the function, formats, and operands of these special commands, see Chapter 4, "Special Commands Used Within Other Commands," on page 1385.

CMS Commands Not Described in This Document

Table 4 on page 20 lists specialized CMS commands and commands for other z/VM components and facilities that must operate in the CMS environment, which are not described in this document.

Table 4. CMS Commands Described in Other Documents

Command	Usage
AUDIT	Starts and stops server security audit trace processing of file pool activity. For use by file pool operators only. See *z/VM: CMS File Pool Planning, Administration, and Operation*.
BACKUP	Starts a backup of the control data while multiple user mode processing continues. For use by repository file pool operators only. See *z/VM: CMS File Pool Planning, Administration, and Operation*.
CPEREPXA	Formats and edits system error records for output. See Environmental Record Editing and Printing Program (EREP): User's Guide (https://www.ibm.com/docs/en/SSLTBW_2.5.0/pdf/ifc1000_v2r5.pdf).
CRR ERASE LU	Erases LU name and transaction program name (TPN) entries from the CRR log name table. (For use by CRR operators only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
CRR ERASE LUWID	Erases CRR log records associated with the specified instance of the logical unit of work ID (LUWID), preventing resync activity from occurring. (For use by CRR operators only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
CRR QUERY LOG	Displays status of CRR log minidisks. (For use by CRR operators only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
CRR QUERY LOGTABLE	Displays the LU name and transaction program name (TPN) for entries in the CRR log name table. (For use by CRR operators only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
CRR QUERY LU	Displays the status of the logical unit of work ID known to the CRR recovery server. (For use by CRR operators only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
CRR QUERY LUWID	Displays the status of the protected resources and conversations involved in a LUWID known to the CRR recovery server. (For use by CRR operators only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.

Table 4. CMS Commands Described in Other Documents (continued)

Command	Usage
CRR RESUME	Cancels the LUWID's suspended status initiated by the CRR SUSPEND command. (For use by CRR operators only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
CRR RESYNC	Provides a response for a protected resource or conversation when an automatic resynchronization has failed, so the resync can continue for the LUWID. (For use by CRR operators only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
CRR SUSPEND	Puts an LUWID into a suspended state. (For use by CRR operators only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
DATASPACE	Makes a directory control directory eligible for use in a data space. (For use by repository file pool operators only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
DDR	Does backup, restore, and copy operations for entire DASD volumes or minidisks. See *z/VM: CP Commands and Utilities Reference*.
DEFBACKUP	Changes the assignment of the file pool control data backup file. (For use by repository file pool operators only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
DELETE ADMINISTRATOR	Removes administrator authority for the specified file pool from the specified user ID. (For use by file pool administrators only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
DELETE LOCK	Releases an explicit lock on a file or repository file pool directory. (For use by repository file pool operators only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
DELETE PUBLIC	Removes the connect authority given to public on the ENROLL PUBLIC command. (For use by repository file pool operators only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
DELETE USER	Removes a user from a repository file pool. (For use by repository file pool operators only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
DIRECTXA	Processes the control statements in a CP directory file. See *z/VM: CP Commands and Utilities Reference*.
DISABLE	Disables a storage group or file space for write access or all access. (For use by repository file pool operators only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
DOSPLI	Compiles DOS PL/I source code under CMS/DOS. See the documentation for the DOS PL/I compiler product.
ENABLE	Reinstates use of a disabled storage group or file space. (For use by repository file pool operators only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
ENROLL ADMINISTRATOR	Adds a file system administrator to the specified file pool. (For use by file pool administrators only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
ENROLL PUBLIC	Gives connect authority for a repository file pool to all users. (For use by repository file pool administrators only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.

Table 4. CMS Commands Described in Other Documents (continued)

Command	Usage
ENROLL USER	Enrolls a user in the specified repository file pool. (For use by repository file pool administrators only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
ERASE LUNAME	Removes the history of all previously forced work from the SFS logs, as well as the transaction program name associated with the LUNAME. (For use by repository file pool operators only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
ETRACE	Starts and stops external tracing. (For use by file pool operators only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
EXPAND	Adds space to a program in object deck form. See *z/VM: VMSES/E Introduction and Reference*.
FCOBOL	Compiles DOS/VS COBOL source code under CMS/DOS. See the documentation for the DOS COBOL compiler product.
FILEPOOL BACKUP	Backs up all data in a user storage group and all associated file pool catalog data. (For use by repository file pool administrators only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
FILEPOOL CLEANUP	Corrects storage group or administration machine problems. (For use by repository file pool administrators only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
FILEPOOL CONTROL BACKUP	Schedules a control data backup for a specified file pool in multiple user mode. (For use by repository file pool administrators only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
FILEPOOL DISABLE	Disables a storage group or file space for write or all access. (For use by repository file pool administrators only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
FILEPOOL ENABLE	Reinstates the use of a disabled storage group or file space. (For use by repository file pool administrator only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
FILEPOOL FILELOAD	Restores one or more base files from a copy of the storage group data created by FILEPOOL BACKUP. (For use by repository file pool administrators only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
FILEPOOL FORMAT AUDIT	Formats the security audit data created by file pool server processing. (For use by file pool administrators only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
FILEPOOL LIST BACKUP	Lists the repository file pool objects and authorizations contained in a user storage group backup file created by the FILEPOOL BACKUP command. (For use by repository file pool administrators only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
FILEPOOL LIST MINIDISK	Lists the repository file pool base files that have at least one data block on a specific repository pool storage group minidisk. (For use by repository file pool administrators only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
FILEPOOL MINIDISK	Adds one or more storage group minidisks to a file pool in multiple user mode. (For use by repository file pool administrators only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.

Table 4. CMS Commands Described in Other Documents (continued)

Command	Usage
FILEPOOL RENAME	Renames an SFS file space from an existing user ID to a new user ID. (For use by repository file pool administrators only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
FILEPOOL RESTORE	Loads the specified file pool storage group from a copy of the storage group data created by FILEPOOL BACKUP processing. (For use by repository file pool operators only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
FILESERV BACKUP	Starts a file pool server in dedicated maintenance mode to back up the *control data*. (For use by the repository file pool server machines only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
FILESERV CRRLOG	Formats and updates the CRR log minidisks. (For use by the CRR server machines only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
FILESERV DEFAUDIT	Adds, changes, or deletes the assignment of the security audit output file for a file pool. (For use by the file pool server machines only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
FILESERV DEFBACKUP	Adds, changes, or deletes the assignment of the control data backup file for the file pool. (For use by the repository file pool server machines only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
FILESERV DEFCRRLOG	Adds, changes or deletes the CRR log minidisks. (For use by the CRR server machines only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
FILESERV FIXCENT	Sets the century date for all your SFS objects in the file pool server machine. (For use by the repository file pool server machines only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
FILESERV GENERATE	Defines and initializes a new server file pool. (For use by the file pool server machines only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
FILESERV LIST	Displays the contents of the file pool catalogs. (For use by the repository file pool server machines only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
FILESERV LOG	Formats and updates the file pool log minidisks. (For use by the file pool server machines only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
FILESERV MINIDISK	Adds one or more minidisks to one or more storage groups in a file pool in dedicated maintenance mode. (For use by the repository file pool server machines only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
FILESERV MOVEUSER	Moves all the file pool data for a user to a different storage group within the same file pool. (For use by the repository file pool server machines only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
FILESERV REGENERATE	Expands the file pool control minidisk. (For use by the repository file pool server machines only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
FILESERV REORG	Deletes unused file pool catalog entries for each user and reorganizes the file pool catalogs to ensure optimum use of catalog data and index space. (For use by the repository file pool server machines only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
FILESERV START	Invokes server processing to support access to a file pool server or recovery server from other virtual machines. (For use by the file pool server machines only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.

Table 4. CMS Commands Described in Other Documents (continued)

Command	Usage
FORCE	Rolls back (backs out) unprepared work and severs the user's connection to the repository file pool server, and commits or rolls back repository file pool prepared work. (For use by repository file pool operators only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
GRANT ADMIN	Gives a user file pool administration authority. (For use by file pool operators only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
GROUP	Builds a GCS configuration file. See *z/VM: Group Control System*.
HCPLDR	Invokes and controls the system loader. See *z/VM: CP Commands and Utilities Reference*.
INSTFPP	Installs optional feature products from optional feature product tapes. See *z/VM: VMSES/E Introduction and Reference*.
IOCP	Uses the Input/Output Configuration Program. See *z/VM: CP Commands and Utilities Reference*.
ITRACE	Stops and starts server internal trace processing of APPC/VM communication-related activities. (For use by file pool operators only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
MODIFY USER	Modifies a user's file space allocation in the Shared File System. (For use by repository file pool administrators only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
OPENVM CREATE DIRECTORY	Creates a new, empty byte file system (BFS) directory. See *z/VM: OpenExtensions Commands Reference*.
OPENVM CREATE EXTLINK	Creates a byte file system (BFS) path name that will reference a file or other data that resides outside the BFS. See *z/VM: OpenExtensions Commands Reference*.
OPENVM CREATE LINK	Creates a new byte file system (BFS) path name to be used to reference another file in the same BFS. See *z/VM: OpenExtensions Commands Reference*.
OPENVM CREATE SYMLINK	Creates a byte file system (BFS) path name to be used to reference an object residing in one BFS using a path name in the same or another BFS. See *z/VM: OpenExtensions Commands Reference*.
OPENVM ERASE	Erases a byte file system (BFS) object. See *z/VM: OpenExtensions Commands Reference*.
OPENVM GETBFS	Copies a byte file system (BFS) regular file into another BFS regular file, an SFS directory, or onto a CMS minidisk. See *z/VM: OpenExtensions Commands Reference*.
OPENVM LISTFILE	Obtains information about files and other objects residing in the displayed byte file system (BFS) directory. See *z/VM: OpenExtensions Commands Reference*.
OPENVM MOUNT	Allows a byte file system (BFS) subdirectory tree or an entire BFS to be logically included as a component of a BFS at any place in the hierarchy. See *z/VM: OpenExtensions Commands Reference*.
OPENVM OWNER	Changes the group or owner (or both) of a byte file system (BFS) object. See *z/VM: OpenExtensions Commands Reference*.
OPENVM PERMIT	Changes the permission bits used to control the owner access, group access, and general access to a byte file system (BFS) object. See *z/VM: OpenExtensions Commands Reference*.

Table 4. CMS Commands Described in Other Documents (continued)

Command	Usage
OPENVM PUTBFS	Copies data into a byte file system (BFS) regular file from another BFS regular file, a file in an SFS directory, or a file on a CMS minidisk. See *z/VM: OpenExtensions Commands Reference*.
OPENVM QUERY DIRECTORY	Displays your current working directory. See *z/VM: OpenExtensions Commands Reference*.
OPENVM QUERY LINK	Displays information associated with symbolic or external links in a byte file system (BFS). See *z/VM: OpenExtensions Commands Reference*.
OPENVM QUERY MASK	Displays the file creation mask values in effect that indicate who will be denied permission to a newly created byte file system (BFS) object. See *z/VM: OpenExtensions Commands Reference*.
OPENVM QUERY MOUNT	Displays what is mounted in your byte file system (BFS) hierarchy. See *z/VM: OpenExtensions Commands Reference*.
OPENVM RENAME	Renames or relocates a byte file system (BFS) object. See *z/VM: OpenExtensions Commands Reference*.
OPENVM RUN	Starts an application that resides in the byte file system (BFS) or has an external link to a CMS module in the record file system. See *z/VM: OpenExtensions Commands Reference*.
OPENVM SET DIRECTORY	Sets or changes your working directory from the current one to a new one. See *z/VM: OpenExtensions Commands Reference*.
OPENVM SET MASK	Defines the file creation mask to be used when creating a byte file system (BFS) object. See *z/VM: OpenExtensions Commands Reference*.
OPENVM SHELL	Starts the OpenExtensions Shell and enters the shell command environment. See *z/VM: OpenExtensions Commands Reference*.
OPENVM UNMOUNT	Removes a byte file system (BFS) or BFS subdirectory tree from your hierarchy. See *z/VM: OpenExtensions Commands Reference*.
PROB	Enters a problem report in the Dump Viewing Facility. See *z/VM: Dump Viewing Facility*.
PROP	Provides Programmable Operator capability. See *z/VM: CMS Planning and Administration*.
QUERY ACCESSORS	Displays information about current accessors of directory control directories, optionally including usage of data spaces. (For use by repository file pool administrators only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
QUERY DATASPACE	Displays information about the eligibility of directory control directories for use of data spaces. (For use by SFS administrators only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
QUERY DEFBACKUP	Determines the default and current assignments of the file pool control data backup file or determines where the last control data backup file was created. (For use by repository file pool operators only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
QUERY DISABLE	Determines if a storage group or file space has been previously disabled and the user ID of the disabler. (For use by repository file pool operators only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.

Table 4. CMS Commands Described in Other Documents (continued)

Command	Usage
QUERY FILEPOOL AGENT	Displays information about the users or internal processes for which the file pool server is currently doing work. (For use by file pool administrators only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
QUERY FILEPOOL CATALOG	Displays information about the file pool catalog space. (For use by file pool administrators only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
QUERY FILEPOOL COUNTER	Displays information about file pool server processing against a file pool. (For use by file pool administrators only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
QUERY FILEPOOL CRR	Displays CRR counter information. (For use by CRR administrators only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
QUERY FILEPOOL LOG	Displays information about the file pool log minidisks. (For use by file pool administrators only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
QUERY FILEPOOL MINIDISK	Displays all minidisk information. (For use by file pool administrators only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
QUERY FILEPOOL OVERVIEW	Displays overview information about a specified file pool. (For use by file pool administrators only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
QUERY FILEPOOL REPORT	Displays information about a specified file pool, and file pool server and recovery server processing against the file pool. (For use by file pool administrators only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
QUERY FILEPOOL STATUS	Displays information about a specified file pool, and file pool server and recovery server processing against the file pool. (For use by file pool administrators only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
QUERY FILEPOOL STORGRP	Displays information on storage groups. (For use by file pool administrators only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
QUERY LIMITS	Displays information about selected users in a particular file pool. (For use by repository file pool administrators only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
QUERY LOGTABLE	Displays the specified LU name and transaction program name (TPN) for entries in the repository file pool log name table. (For use by repository file pool operators only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
QUERY PREPARED	Displays information about SFS prepared work and forced work. (For use by repository file pool operators only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
REVOKE ADMIN	Deletes file pool administration authority from a user. (For use by file pool operators only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
SET THRESHOLD	Sets a threshold limit on your own or another user's file space which defines when a warning message and return code should be issued that indicates a certain amount of allocated file space is used. (For use by repository file pool operators only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
SNTINFO	Gets discontiguous saved segment (DCSS) information directly from CP. See *z/VM: VMSES/E Introduction and Reference*.

Table 4. CMS Commands Described in Other Documents (continued)

Command	Usage
SNTMAP	Processes the macro definitions in a system name table (SNT) file. See *z/VM: VMSES/E Introduction and Reference*.
STAT	Displays the status of reported system problems. See *z/VM: Dump Viewing Facility*.
STOP	Stops multiple user mode processing for a file pool. (For use by file pool operators only.) See *z/VM: CMS File Pool Planning, Administration, and Operation*.
UTILITY	Does installation and service utility tasks: obtains DASD information; creates a stand-alone service utility tape including either or both of the following stand-alone utility programs: ICKDSF DDRXA. See *z/VM: CP Commands and Utilities Reference*.
VMFMERGE	Applies PTFs to SNA products. See *z/VM: VMSES/E Introduction and Reference*.
VMFREMOV	Removes PTFs from SNA products. See *z/VM: VMSES/E Introduction and Reference*.
VMFZAP	Applies ZAPs to SNA products. See *z/VM: VMSES/E Introduction and Reference*.
ZAPTEXT	Modifies or dumps individual text files. See *z/VM: VMSES/E Introduction and Reference*.

Note: All commands beginning with "VMF" that are not documented in this book or listed in this table, including those which were formerly CMS commands, are VMSES/E commands. They are documented in *z/VM: VMSES/E Introduction and Reference*.

Chapter 2. CMS Commands

This section contains reference information on CMS commands. Most of these commands are intended for general users. The commands are listed in alphabetical order.

For information on the CMS utilities, see Chapter 3, "Utilities," on page 1291.

Some specialized CMS commands and commands for other z/VM components and facilities that operate in the CMS environment are not described in this document. See "CMS Commands Not Described in This Document" on page 20.

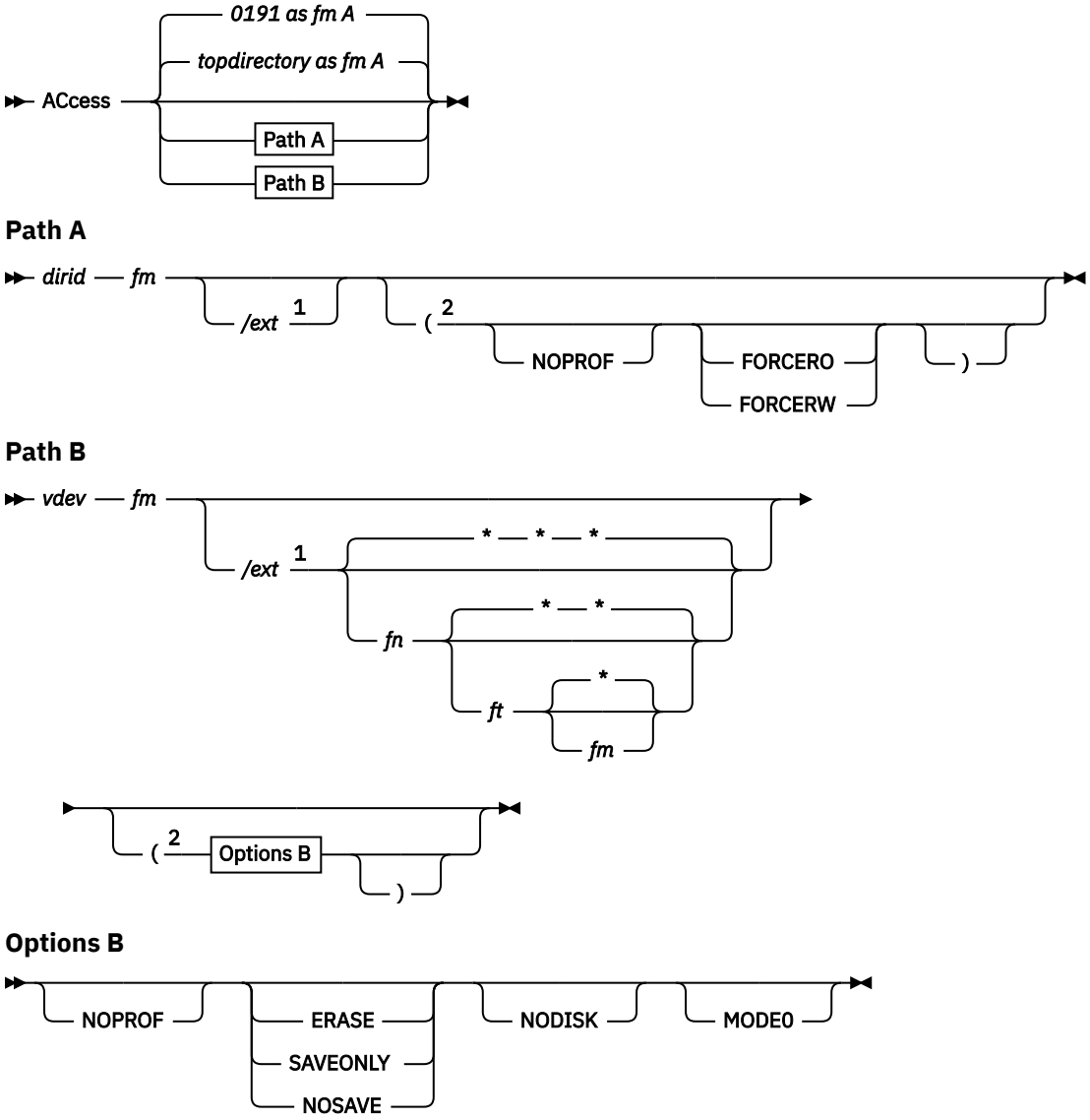
The following commands and subcommands exist in multiple environments:

Command	Environments
CP	CP, CMS, XEDIT
HELP	CMS, XEDIT
LOAD	CMS, XEDIT
QUERY	CP, CMS, XEDIT
RESET	CP, XEDIT
SET	CP, CMS, XEDIT
SORT	CMS, XEDIT
XEDIT	CMS, XEDIT

Each command description is presented in the following format (some sections might not be included):

- ***Name***: Identifies the name of the command.
- ***Format***: Shows the syntax of the command with all the possible operands and options you can use.
- ***Authorization***: Identifies the type of user who can issue the command.
- ***Purpose***: States what the command is used for.
- ***Operands and Options***: Defines the function of each operand and option and any values you can include.
- ***Usage Notes***: Identifies and describes special situations and other considerations that may affect your use of the command.
- ***Examples***: Provides one or more examples to show how the command is commonly used.
- ***Responses***: Describes the responses you might receive from the command on your display device. Responses are normal operational output; they tell you about the execution and effect of the command. Unlike system messages, command responses are not prefixed with an identifying number and are not contained in *z/VM: CMS and REXX/VM Messages and Codes*.
- ***Messages and Return Codes***: Lists the messages issued by the command. Messages not unique to the command might also be issued. Each message is prefixed with an identifying number. For more information on the messages, including suggested actions, see *z/VM: CMS and REXX/VM Messages and Codes*.

ACCESS



Notes:

[1] Do not specify a space between *fm* and */ext*.
[2] You can enter Options in any order between the parentheses.

Authorization

General User

Purpose

Use the ACCESS command to:

- Identify a minidisk or Shared File System (SFS) directory to CMS.
- Make a list of the files on the specified minidisk or directory available to your virtual machine.

- Establish a file mode letter for the files on a minidisk or in a directory.

Operands

0191
topdirectory
If you issue ACCESS without any operands, the 0191 disk or top directory is accessed as file mode A. If both the 0191 disk and top directory exist, the top directory will be the one to be 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 information on *dirid*, see “Naming Shared File System (SFS) Directories” on page 6.

fm
assigns a one-character file mode letter to all files on the minidisk

/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 number. The valid numbers are X'0001' through X'FFFF'.

fn
ft
fm
define 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. For more information, see Usage Notes “2” on page 37 and “3” on page 37, "Using the ACCESS command with *vdev*". To specify a file mode, use a letter and a number, for example:

```
B1
```

The allowable file mode numbers are 0 to 6. For more information about file mode 0 files see, Usage Note “3” on page 37, "Using the ACCESS Command with *vdev*".

For more information about OS and DOS disk access restrictions, see Usage Note “8” on page 37, "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.

FORCERO
forces the SFS directory to be accessed in read-only status. You do not need authority to the files in the directory, just to the directory itself. For a file control directory, you need either READ or WRITE authority. For a directory control directory, you need DIRREAD authority.

FORCERW
forces the SFS directory to be accessed in read/write status. You can specify FORCERW even if you have only read authority to the directory. For directories with the (DIRCONTROL) attribute, directory control write (DIRWRITE) authority is required. This option is not valid for SFS directories accessed as extensions to another file mode.

ERASE
indicates you want to erase all the files on the specified minidisk. This option is only valid for read/write disks. If the minidisk is read-only, you will receive the following message: `DMS003E Invalid option: ERASE`. For more information, see Usage Note “6” on page 37, "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.

NODISK
lets you gain access to the CMS operating system with no minidisks 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.

MODE0
accesses the minidisk and includes file mode 0 files. This option is meaningful only to minidisks linked read-only. The ACCESSM0 command must be set to ON before you can use the MODE0 option of the ACCESS command.

Usage Notes

Using the ACCESS Command with a Directory ID or a Device Number

1. When you IPL CMS, if you have a default file pool defined in your z/VM directory, or if you define it when you IPL CMS, your top directory in that file pool is accessed as file mode A. If no file pool is defined and you have a defined disk number of 0191 in your z/VM directory, or you define it before you IPL CMS, your 0191 disk is accessed as A. If you do not have either a top directory in the default file pool or a 0191 disk defined, nothing is accessed as file mode A.

   Issuing ACCESS without any parameters or options accesses your top directory or 0191 disk in the same manner.
2. If you have defined disk numbers 190 and 19E in the z/VM directory, or if they are defined before you IPL CMS, these disks are accessed as file mode S and Y respectively. If you have a minidisk defined as virtual device number 192, it may be automatically accessed as D when you IPL CMS:
   - If 192 is an unformatted temporary minidisk or virtual disk in storage, CMS formats it and accesses it as file mode D.
   - If 192 is a CP-formatted temporary minidisk or virtual disk in storage, CMS reformats it for CMS and accesses it as file mode D.
   - If 192 is a CMS-formatted temporary minidisk, virtual disk in storage, or permanent minidisk accessed as a file mode other than D, CMS reaccesses it as file mode D.
   - If 192 is an unformatted or CP-formatted permanent minidisk, CMS does not automatically format, reformat, access, or reaccess it.

   When CMS accesses 190 as S, 19E as Y, or 192 as D, any minidisk or SFS directory already accessed as that file mode is released.

   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: 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. You must issue another ACCESS command 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 0191 a/a
   ```

   forces your 0191 disk into read-only status.

   Note: Having a directory in read-only status does not restrict all writing to the directory or files. Use the CREATE LOCK command to protect SFS files and directories.

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 more information on the read-only extensions, see *z/VM: CMS User's Guide*.

6. When you access a minidisk, a list of the files on the minidisk 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.

   That is, when another user creates, erases, or updates files on the minidisk, the minidisk's file directory is updated to reflect the current status of the minidisk, but copies in other users' storage is not updated to reflect the changes. Users that have the minidisk accessed read-only, must re-access the minidisk to see any changes. Thus, if you have a minidisk accessed read-only while another user is writing to a file, you may need to periodically reenter the ACCESS command for the minidisk to obtain a fresh copy of the file. You will see the updates to the file up to the last save.

   When you access a file control directory, a list of files is also stored in your virtual machine. Unlike minidisks, however, the list of files is updated automatically for you. There is no need for you to reaccess the directory when users create, erase, or change files.

   When you access a directory control directory, a list of files is sometimes stored in your virtual storage. If you are using an XC virtual machine and the directory is in a data space, CMS puts the list of files in the data space. Regardless of where the list of files is maintained, however, directory control directories work like minidisks. That is, when another user creates, erases, or updates files in the directory, the data in the file pool is updated, but the version of the directory you have accessed is not. You must reaccess the directory to see any changes.

7. If you have many directories or minidisks accessed and you are experiencing poor response time, release the ones you are not using. For more information, see “RELEASE” on page 815.

8. If you are accessing minidisks and SFS directories that contain many files or you are accessing many minidisks or SFS directories, you may get a virtual storage exceeded error. This is because a list of files is stored in your virtual machine.

   You can increase the storage of your virtual machine and re-ipl. If the problem still occurs, you must free some virtual storage.

   a. Release some minidisks or SFS directories that are no longer needed.
   b. Reorganize the files. Break up a minidisk or SFS directory with a large number of files into multiple minidisks and directories.
   c. Reduce the number of files.

9. When a directory control directory is accessed (read-only) more than once without first releasing it, all nested accesses must be released before committed changes become available to the user (upon next access).

10. You cannot use the ACCESS command to access minidisks formatted in 800-byte blocks.

11. Because the CMS file system requires file status and control information to reside below 16 MB in virtual storage, there is a practical limitation on the size of CMS minidisks. As a minidisk increases in size, or more files reside on the disk, the amount of virtual storage associated with the disk for CMS file system status and control increases in storage below 16 MB. The current ECKD DASD limitation is 65520 cylinders for a 3390 disk on an IBM TotalStorage DASD subsystem, or about 45 GB of data. The maximum size for FBA SCSI disks supported for use by CMS or GCS is 381 GB. IBM suggests that customers defining disks for use by CMS should set a practical limit of about 22 GB. If larger disks are

defined, they should be limited to contain very few files, or the CMS file system may not be able to obtain enough virtual storage below 16 MB to format or access those disks.

Using the ACCESS Command with a Directory ID

1. When you access an SFS directory without forcing the status to read-only or read/write, the status is determined by who owns it.
   - If you are the directory owner, it is accessed as read/write. For directory control directories, however, only one user can have the directory accessed in read/write status. If someone already has the directory accessed in read/write status, and you (the directory owner) attempt to get a read/write access, you will, instead, get a read-only access.
   - If you are not the directory owner, 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. To force a directory you own to be accessed in read-only status, you can either:
   - Access the directory as an extension of another minidisk, directory, or of itself. For example:

     ```
     access vmsysu:yourid.test a/a
     ```

   - Specify the FORCERO option:

     ```
     access vmsysu:yourid.test a (forcero
     ```

   Any directory you do not own is, by default, accessed in read-only status, even if you have write authority to that directory.
3. When you access an SFS directory in read-only status, you can write to a file in that directory if all the following are true:
   - The directory has the file control attribute.
   - You have WRITE authority to the file.
   - The CMS command you are using to write to the file does not require the directory to be accessed read/write. (CSL routines do not require the directory to be accessed read/write, but many CMS commands do.)

   For more information on the CMS commands that do not require a directory to be accessed R/W, see *z/VM: CMS User's Guide*.

   You cannot write to a file in a directory control directory if you have that directory accessed in read-only status. The attempts will fail even if you own the directory or if you have directory control write (DIRWRITE) authority to it.

   To write to file in a directory control directory, access the directory in read/write status. You can also write to files by not accessing the directory and, instead, using CMS commands or Callable Services Library (CSL) routines that let you specify a directory name.
4. If you are not the directory owner, but have some authority to the directory, you may be able to force the directory into read/write status. For file control directories, you can force any directory into read/write status so long as you have write or *read* authority to that directory. For directory control directories, you must have directory control write (DIRWRITE) authority to force the directory into read/write status. Directory control directories also have the restriction that only one user at a time may have the directory accessed in read/write status. So, you may not always be able to force a directory control directory into read/write status, even if you are properly authorized.

   To force a directory into read/write status, use the FORCERW option:

   ```
   access vmsysu:dave.notes b (forcerw
   ```

   You cannot specify FORCERW if you are accessing the directory as an extension of some other file mode.
5. If you force a file control directory into read/write status you can use commands that require a read/write file mode. Of course, you must have proper authority for the files you use.

Suppose, for example, you have write authority to the PROJECT NOTEBOOK file in the VMSYSU:MINDY.SERVICE directory. You want to use EXECIO to write to the file, which requires a read/write file mode. However, you have only read authority to the directory. You can access Mindy's directory in read/write status using the FORCERW option:

```
access vmsysu:mindy.service b (forcerw
```

Now you can write to the PROJECT NOTEBOOK file. Because the file exists and you have write authority, almost any command that writes to it will succeed. If, however, the command internally creates a temporary file, it will fail because you do not have write authority to the directory. You will see messages similar to these:

```
DMS105S Error nn writing file fn ft fm on disk or directory
DMS1258E You are not authorized to write to file fn ft fm
```

Any command that tries to create a new file in the directory will fail for the same reason.

Applications or execs may also fail for similar reasons. Many applications try to create new files or temporary files on read/write file modes. These applications will fail if you have only read authority to the directory.

6. If you have write authority to another user's directory and specify FORCERW, your applications or commands may conflict with those of other authorized users.

   Suppose, for example, you use FORCERW to access a directory owned by user DAVE. Dave has granted you write authority on his directory. Now suppose you run an application that creates a work file in Dave's directory. Meanwhile, Dave runs the same application using the same directory. Dave's application will see the work file you created, but replaces it anyway because it is only a work file. Your application continues execution and now reads the work file which, of course, contains the wrong data. Or, if the application erases the work file instead of replacing its contents, your authorization to the file will be lost when Dave runs the application. In either case, you get undesired results. So, be careful when using FORCERW -it is intended for special use.
7. If a file control directory is locked EXCLUSIVE, only the holder of the lock can access the directory. If a file control directory is locked SHARE or UPDATE, any authorized user can access the directory.
8. You can access a file control directory in read-only mode even if it has no files in it. You cannot access a directory control directory in read-only mode if it does not contain files or subdirectories.

Using the ACCESS Command with DIRCONTROL Directories

This section contains notes that apply only to SFS directories with the directory control attribute. Use the QUERY DIRATTR command to determine whether the directory attribute is FILECONTROL or DIRCONTROL.

1. To access a directory with the DIRCONTROL attribute, you must have directory control read (DIRREAD) or directory control write (DIRWRITE) authority, depending on the mode of access. Users who create (own) directory control directories automatically have DIRREAD and DIRWRITE authority for those directories. File pool administrators have implicit DIRREAD and DIRWRITE authority to every directory control directory in the file pool.

   A directory accessed with the DIRCONTROL attribute has unique functional characteristics. These characteristics are described in the CREATE DIRECTORY command.

   Some of these special characteristics involve the ACCESS command itself. The Table 5 on page 35 shows the specific conditions and results for accessing a directory with the DIRCONTROL attribute.

Table 5. Accessing a Directory with the Directory Control Attribute

ACCESS		DIRECTORY CONTROL AUTHORITY		
Force R/O	**Force R/W**	**Owner**	**Dirread**	**Dirwrite**
—	—	Directory Control Write (Note a)	Directory Control Read	Directory Control Read

Table 5. Accessing a Directory with the Directory Control Attribute (continued)

ACCESS		DIRECTORY CONTROL AUTHORITY		
Force R/O	**Force R/W**	**Owner**	**Dirread**	**Dirwrite**
—	X	Directory Control Write (Note c)	Deny Access	Directory Control Write (Note b)
X	—	Directory Control Read	Directory Control Read	Directory Control Read

Notes on the Table:

a. When the owner accesses a directory control directory without specifying FORCERO or FORCERW, the owner usually gets a read/write access. The owner will get a read-only access in the following cases:

- The directory is already accessed in read/write mode by another user.
- The owner or another user has an UPDATE lock on the directory.
- Another user has an UPDATE lock on a file in the directory.

Also, if the same user accesses the directory again, the user's prior accesses will be released *except* when the latest access results in one of these two conditions:

- The directory is being accessed read-only at more than one file mode (each access is, therefore, for the same version of the directory).
- The directory is already accessed read/write and the latest access is for read-only at a different file mode (this access results in an error).

b. When a nonowner accesses a directory control directory using the FORCERW option and someone already had it accessed or open in read/write mode, or locked in UPDATE mode, the request is denied.

c. When an owner accesses a directory control directory using FORCERW and the directory is already accessed or open in read/write mode, the request is denied.

2. Only one user at a time can have a directory control directory accessed in read/write mode. Attempts to access a directory control directory in read/write status (using the FORCERW option) will fail if the directory is already accessed in read/write mode by another user.

   An attempt to access a directory control directory in read/write mode will also fail if another user has locked a file in the directory. In this case, an attempt to access the directory in read-only mode will succeed.

3. When another user has a directory accessed in read/write mode, you cannot change files in that directory, even if you are authorized to do so.

4. It is possible for a user to write directly to a file in an SFS directory without accessing the directory. (This can be done with Callable Services Library routines in application programs.) If you try to access the directory in read/write mode while someone is directly writing to a file in the directory, your access will wait or be rejected according to the setting of the FILEWAIT option. For more information, see the “SET FILEWAIT” on page 933.

5. You cannot make changes to files or create subdirectories in a directory control directory you have accessed in read-only mode.

6. Once you have a directory control directory accessed in read-only status, you will not see any changes in that directory (including the contents of the files) that are made by other users. To see the changes, you must access the directory again. At that time, you will be able to see any committed changes.

   However, if you are the writer, you do not need to reaccess in order to see the changes. They are always available to you.

   Likewise, if a directory control directory is renamed or relocated while you have it accessed read-only, you must release the directory before you can access it under the new name.

Exception: When a file has the INPLACE attribute, you see changes as they are written to the file pool. You do not need to wait until the changes are committed, and you do not need to reaccess the directory. For more information about the INPLACE files, see *z/VM: CMS Application Development Guide*.

Using the ACCESS Command with a Virtual Device Number

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 NOSAVE options accesses the minidisk using a saved copy of the file directory that contains entries for only those files you can access. 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 you obtain the appropriate file directory.
2. The *fn ft fm* fields can only be specified for minidisks 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 number 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 number 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 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 virtual number 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. To see OS and DOS disks, you must have a mode A, if you are going to use the LOAD command with the MAP option. The default option is MAP.
c. If two or more minidisks have been accessed in CMS, and CP DEFINE commands are executed that swap virtual numbers, 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
```

Note: The 193 disk is the MAINT 193 disk.

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

9. Using the MODE0 Option
   - If MODE0 is specified for a disk accessed read/write, or for a disk accessed read-only that is linked read/write, file mode 0 files will be brought into storage regardless of whether the ACCESSM0 command has been previously set to ON.
   - If you specify MODE0 for a disk accessed read-only that is linked read-only, file mode 0 files will be included only when the MODE0 option has been enabled by previously setting the ACCESSM0 command to ON.
   - Specifying file mode number 0 for ACCESS has the same effect as specifying MODE0. For example:

     ```
     access 391 d/d * * d0
     ```

     is equivalent to:

     ```
     access 391 d/d * * d0 (MODE0
     ```

     The same rules that apply for file mode 0 files apply whether the option is specified or implied.
   - If MODE0 is specified and the file mode number (if specified) is not zero, an error will be issued.
   - If MODE0 is specified, nonsaved storage access is performed (that is, the file directory is brought into your virtual storage, and the saved copy of the file directory in a shared segment is not used). This is because saved storage access will not have file mode 0 files saved in the segment.

Examples

Assuming 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 number 498 as file mode B, enter the following:

```
access 498 b
```

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

```
access 498 b (saveonly
```

For more examples on how to use the ACCESS command, see *z/VM: CMS User's Guide*.

Responses

DMS723I *mode* (*vdev* | *dirname*) {R/O|R/W} [-OS|-DOS]
This message is displayed if the specified minidisk 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.

DMS724I *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.

DMS725I *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.

DMS726I *vdev* | *dirname mode* released
The minidisk being accessed at virtual number vdev as a read/write minidisk, or the directory dirname, is already accessed at a different mode. It is released from that mode. This message is also displayed when directory control directories are reaccessed.

Messages and Return Codes

- DMS003E Invalid option *option* [RC=24]
- DMS017E Invalid device address *vdev* [RC=24]
- DMS048E Invalid filemode *mode* [RC=24]
- DMS059E *vdev* | *dirname* already accessed as read/write filemode *fm* [RC=36]
- DMS060E File [*fn* [*ft* [*fm*]]] not found; filemode *fm*[*vdev* | *dirid*] will not be accessed [RC=28]
- DMS066E *option1* and *option2* are conflicting options [RC=24]
- DMS070E Invalid parameter *parameter* [RC=24]
- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS112S Disk *fm*(*vdev*) device error [RC=100]
- DMS113S *fm*(*vdev*) not attached or invalid device address [RC=100]
- DMS114S Device *vdev* too large for CMS use [RC=100]
- DMS230W O/S disk-fileid and options specified are ignored [RC=4]
- DMS260E Disk not properly formatted for ACCESS [RC=88]
- DMS724I *vdev* replaces *fm (vdev)*
- DMS1000E The accessing of file mode 0 files must be enabled by prior use of the ACCESSM0 command RC=24]
- DMS1078E Cannot access saved file directory for this disk [RC=44]
- DMS1153E File pool unavailable [RC=99]
- DMS1184E Directory *dirname* not found or you are not authorized for it [RC=100]
- DMS1198E {File|Directory} [*fn ft*] *fm*|*dirname* is currently open; it must be closed before it can be accessed. [RC=70]
- DMS1216E Option *option* is not valid when used for a directory [RC=24]
- DMS1216E Parameter *parameter* is not valid when used for a file in a directory [RC=24]
- DMS1223E There is no default file pool currently defined [RC=40]
- DMS1233E Invalid use of FORCERO/FORCERW option [RC=24]
- DMS1240E You are not authorized to connect to file pool *filepoolid* RC=76|31]
- DMS2133E *bfsid* is a byte file system. It cannot be accessed [RC=24]
- DMS2522E DIRCONTROL directory *dirid2* is already accessed using previous directory name *dirid1* [RC=36]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411
Errors in the Shared File System	“File Pool Server Messages” on page 1414

ACCESSM0

Authorization

General User

Purpose

Use the ACCESSM0 command to indicate whether file mode 0 files can be brought into storage if the MODE0 option of the ACCESS command is specified or implied.

Operands

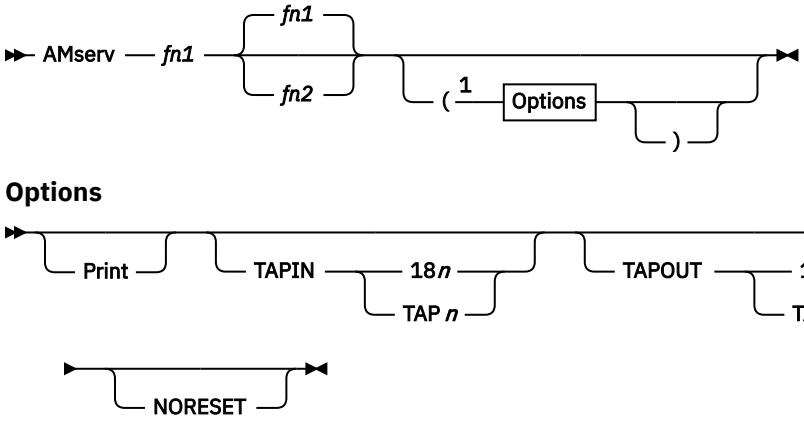
ON
allows read-only access to file mode 0 files.

OFF
does not allow read-only access to file mode 0 files.

Usage Notes

1. When you IPL CMS, ACCESSM0 is always OFF.
2. The ACCESSM0 setting is in effect until the CMS session ends. A CMS session ends with system reset, re-IPL, or log off.
3. The command can be explicitly issued by a user, exec, or application, or in a profile or system profile.
4. This command is an installation option and is initially available only to users accessing the MAINT 193 disk. If the system does not recognize this command, you should consult your system administrator. It may be desirable to copy the command module onto a shared or system disk.

AMSERV



Notes:

[1] You can enter Options in any order between the parentheses.

Authorization

General User

Purpose

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.

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 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 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 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.

NORESET
specifies the call to access method services is being made dynamically by an application program and the VSAM environment will be left intact after access method services has completed. This option is most useful when the AMSERV command is issued from an exec or application program.

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. The restrictions placed on VSAM usage in CMS are listed in *z/VM: CMS Application Development Guide for Assembler*. For more information on access method services control statements format and syntax, see *Using VSE/VSAM Commands and Macros*.
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 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 stops or you issue HX to stop an AMSERV command, the AMSERV environment may not be reset correctly. If a subsequent AMSERV command 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.
9. When the NORESET option is specified, the VSAM environment will be left intact until one of the following occurs:
   - An EXECOS command is issued.
   - An abend occurs.
   - The `Ready;` message appears.
   - Another AMSERV command is issued without the NORESET option.
   - The CATCHECK command is issued.
   - A DOS EOJ (SVC 14) or CANCEL (SVC 6) is executed.
   - Completion of the OS/VSAM program call within an exec.
   - A SET DOS OFF command is issued.

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 in effect when you invoke the AMSERV command are saved and restored when the command completes executing.

Examples

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 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.

DMS367R Enter tape {input|output} DDNAMEs:
This message prompts you to enter the ddnames associated with the tape files.

DMS722I 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

- DMS001E No filename specified [RC=24]
- DMS002E File FNAME1 AMSERV not found [RC=28]
- DMS003E Invalid option: *option* [RC=24]
- DMS006E No read/write filemode accessed for FNAME2 LISTING [RC=36]
- DMS007E File FNAME1 AMSERV *fm* not fixed, 80-character records [RC=32]
- DMS065E *option* option specified twice [RC=24]
- DMS066E *option1* and *option2* are conflicting options [RC=24]
- DMS070E Invalid parameter *parameter* [RC=24]
- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS113S Tap*n*(*vdev*) not attached [RC=100]
- DMS136S Unable to load IDCAMS [RC=104]
- DMS228E No DDNAME entered [RC=24]
- DMS283E The *name* saved segment could not be found; return code *cc* from SEGMENT [RC=128]
- DMS400S System *sysname* does not exist [RC=44]

ASMGEND

Authorization

Installer

Purpose

Use the ASMGEND EXEC to build the system assembler and create the associated auxiliary directory. ASMGEND loads the text decks for the assembler in the correct overlay structure and produces a load map, then generates the ASSEMBLE MODULE.

Usage Notes

The assembler text decks generally reside on the CMS system minidisk (MAINT 190) in file mode S1. ASMGEND writes the new ASSEMBLE module on file mode A. Therefore, access the system minidisk as file mode A before entering the ASMGEND command. Enter:

```
access 190 a
vmfsetup ppfname cms
asmgend
ipl 190 clear parm savesys cms
```

Responses

```
ENTER TARGET DISK MODE FOR ASSEMBLE MODULES
DEFAULTS TO S-DISK IF NONE ENTERED
```

Enter the file mode letter of the minidisk containing the assembler modules. This should be the original accessed mode and not the additional file mode accessed (see usage notes). The ASMGEND command accesses this minidisk during processing. If you enter a file mode letter, ASMGEND uses that file mode letter as the *targetmode* operand of the GENDIRT command when it creates the auxiliary directory. If you do not specify a file mode letter, S is used.

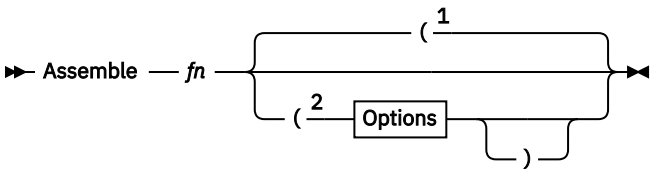
```
ASSEMBLE XF GEND COMPLETE
```

The system assembler and its associated auxiliary directory have been generated successfully.

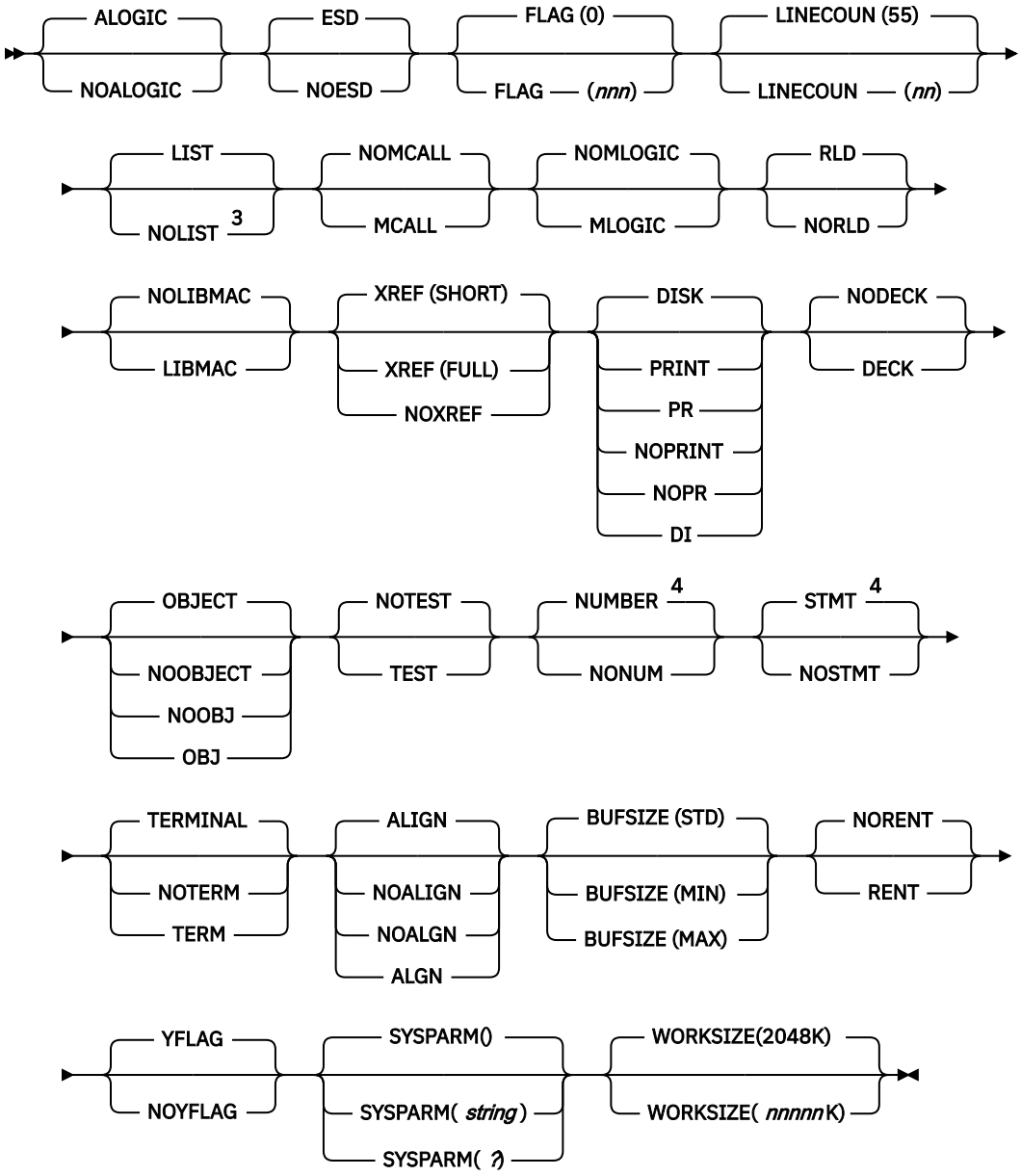
```
ASSEMBLE XF GEND FAILED
```

The system assembler text files were not loaded successfully.

ASSEMBLE



Options



Notes:

[1] The defaults you receive appear above the line in the Options fragment.
[2] You can enter Options in any order between the parentheses.
[3] NOLIST overrides ESD, RLD, and XREF options.
[4] NUMBER and STMT are only valid with the TERM option.

Authorization

General User

Purpose

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

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 system assembler expects a CMS file with a file type of ASSEMBLE.

Options

Listing Control Options

This list describes the system assembler options you can use to control the assembler listing.

ALOGIC
lists conditional assembly statements in open code. This is the default.

NOALOGIC
suppresses the ALOGIC option.

ESD
lists the external symbol dictionary (ESD). This is the default.

NOESD
suppresses the printing of the ESD listing.

FLAG *(nnn)*
does not include diagnostic messages and MNOTE messages below severity code *nnn* in the listing. The default is 0. 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)*
nn specifies the number of lines to be listed per page. The default is 55. For more information, see Usage Note “9” on page 52.

LIST
produces an assembler listing. Any previous listing is erased. This is the default.

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 macroinstructions encountered during macro generation following their respective outer macroinstructions. The system 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. This is the default.

MLOGIC
lists all statements of a macrodefinition processed during macrogeneration after the macroinstruction. The system assembler assigns statement numbers to them.

NOMLOGIC
suppresses the MLOGIC option. This is the default.

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

NORLD
does not print the relocation dictionary.

LIBMAC
lists the macrodefinitions 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 macrogeneration. It may appear in a macro or in open code; it may even be created by substitution. The system assembler assigns statement numbers to the statements that follow the logical END statement.

NOLIBMAC
suppresses the LIBMAC option. This is the default.

XREF
may be:

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

(SHORT)
includes in the assembler listing a cross-reference table of all symbols 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. The default is SHORT.

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
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 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, an error message is generated. The default is DISK.

Output Control Options

The output control options are used to control the object module output of the system 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. This is the default.

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. This is the default.

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. This is the default.

SYSTERM Control Options

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

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

NONUM
suppresses the NUMBER option.

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

NOSTMT
suppresses the STMT option.

TERMINAL
TERM
writes the diagnostic information on the SYSTERM data set. The diagnostic information consists of the diagnosed statement followed by the error message issued. This is the default.

NOTERM
suppresses the TERMINAL option.

Other Assembler Options

These options allow you to specify various functions and values for the system 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. This is the default.

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

BUFSIZE (MIN)
may be:

(UT)
uses the minimum buffer sizes (790 bytes) for each of the utility data sets (SYSUT1, SYSUT2, and SYSUT3). The storage generally is used for the buffers 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.

(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 macrogeneration dictionaries. The default is STD.

(MAX)
is useful when many macros or large macros are used in an assembly. The system assembler uses up to 15 save areas for input records and saves the areas according to their frequency of use, optimizing the macrogeneration 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.

RENT
checks your program for a possible violation of program reenterability. Code that makes your program nonreenterable is identified by an error message.

NORENT
suppresses the RENT option. This is the default.

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

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

SYSPARM
passes a character value to the system variable symbol, SYSPARM. You can enter the variable that goes with SYSPARM three different ways:

(*string*)
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.

(?)
Entering SYSPARM (?) causes CMS to prompt you with the message:

```
ENTER SYSPARM:
```

You can enter up to 100 characters.

()
enters a null string of characters. This is the default.

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

WORKSIZE
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 default is 2048K. 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 system 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 DMSGPI MACLIB * (RECFM FB LRECL 80 BLOCK 800
FILEDEF SYSUT1 DISK fn SYSUT1 fm4 (BLOCK 13030 AUXPROC asmproc
FILEDEF SYSUT2 DISK fn SYSUT2 fm4 (BLOCK 13030 AUXPROC asmproc
FILEDEF SYSUT3 DISK fn SYSUT3 fm4 (BLOCK 13030 AUXPROC asmproc
FILEDEF SYSTERM TERMINAL (AUXPROC termproc
```

At the completion of the ASSEMBLE command, all FILEDEFs created by this command are cleared.

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 dmsgpi dmsom osmacro testlib
   ```

   identifies the MACLIB files named DMSGPI, DMSOM, 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
   ```

   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. For more information, see *z/VM: CMS Application Development Guide*.

5. You need not make any logical assignments for input or output files when you use the system assembler under CMS/DOS. For more information on the file definitions assigned by default under CMS, see Usage Note “1” on page 50.

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. For more information on how an assembler language program can supply an extended plist, see *z/VM: CMS Macros and Functions Reference*.

7. When assembling large files, you may want to issue the FILEDEF command for the SYSUT*n* data sets to avoid 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 13030 and a variable record format, which uses less space.
8. If you have assigned default file definitions, the system assembler puts the TEXT file on a virtual disk, searching for the disk in the same manner described in the DISK Listing Control Option.
9. If the value of the LPP option on the CP SPOOL command is 0 (LPP OFF), the default LINECOUN value is 55. If the LPP value is greater than 0, the LINECOUN value will be set to the LPP value minus the number of header lines output by the assembler. If the LPP value minus the number of header lines output by the assembler is greater than 99, the LINECOUN value will be set to 99.

   If the LINECOUN option is specified with the command, the value of LINECOUN overrides the value set by the LPP option. For more information, see *z/VM: CP Commands and Utilities Reference*.

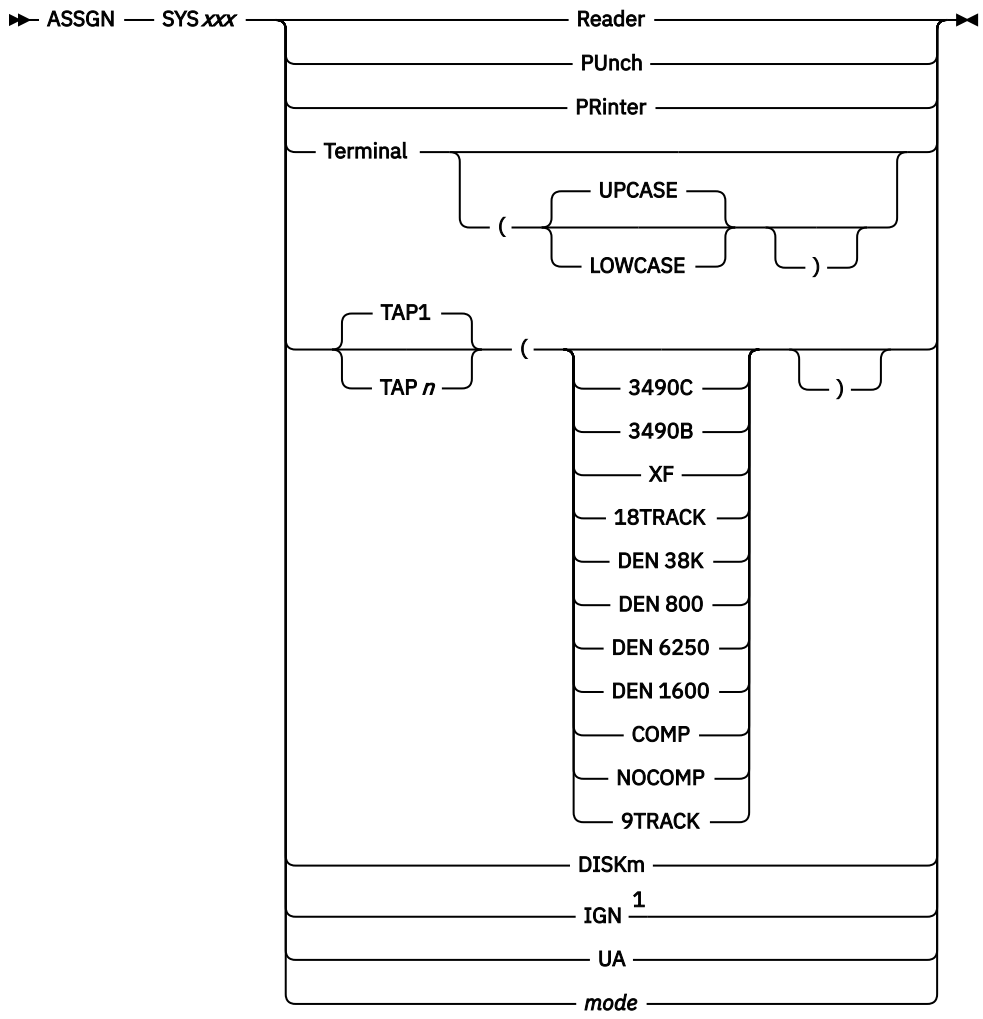
Examples

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.

ASSGN



Notes:

[1] IGN is not valid for SYSRDR, SYSIPT, SYSIN, and SYSCLB.

Authorization

General User

Purpose

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

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:

SYSxxx
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
is a tape device. TAP*n* is one of the following device names for a virtual tape device: TAP1, TAP2, TAP3, or TAP4. (These represent virtual addresses 181, 182, 183, and 184, respectively.) If *n* is omitted, TAP1 is assumed.

CMS defines other tape devices (for example, TAP5) but CMS/DOS only accepts these.

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

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

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.

Note: When using *mode*, you cannot use 'T' and 'R' file modes for assignment of system or programmer logical units since this is restricted to terminal and reader, respectively. In this case, use DISKm.

DISKm
specifies the one-character mode letter of the disk or directory being assigned to the logical unit (SYS*xxx*). 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.

Note: Using DISKm, you can assign system or programmer logical units to 'T' and 'R' file modes.

Options

UPCASE
translates all terminal input data to uppercase. This is the default.

LOWCASE
retains all terminal input data as keyed in.

3490C
specifies 3490 Compacted recording format.

3490B
specifies 3490 Basic recording format.

XF
specifies 3480 Compacted recording format.

18TRACK
specifies 3480 Basic recording format.

DEN 38K
specifies 3480 Basic recording format.

DEN 800
specifies NRZI recording format.

DEN 6250
specifies GCR recording format.

DEN 1600
specifies PE recording format.

COMP
specifies any compacted recording format. CMS selects a compacted recording format for you that the device is capable of writing (if there is one). If you require a particular compacted recording format, use the 3490C or XF option.

NOCOMP
specifies any uncompacted recording format. CMS selects an uncompacted recording format for you that the device is capable of writing (if there is one). If you require a particular uncompacted recording format, use the 3490B, 18TRACK, DEN 6250, DEN 1600, or DEN 800 option.

9TRACK
specifies any 9-track recording format. CMS selects a 9 track recording format for you that the device is capable of writing (if there is one). If you require a particular 9-track recording format, use the DEN 800, DEN 1600, or DEN 6250 option.

Usage Notes

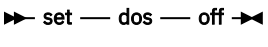
1. When you enter the CMS/DOS environment with the command SET DOS ON, SYSLOG is assigned by default to TERMINAL. If you specify the mode letter of the VSE system residence on the SET DOS ON command line, SYSRES is assigned to that mode.
2. You cannot assign any of the following VSE system logical units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

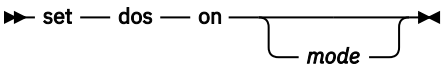
```
SYSRES  SYSLNK  SYSDMP
SYSCTL  SYSREC
```

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. When you make an assignment to a tape or to the terminal, any options you specify or allow to default pertain to the device and not to the specific logical unit. Thus, if you assign more than one logical unit to the same tape device or to the terminal, the options you specify or allow to default will be in effect for all logical units assigned to that device. For example, if you do the following:

   ```
   ASSGN SYS001 TERMINAL ( LOWCASE
   ASSGN SYS002 TERMINAL ( UPCASE
   ```

   the UPCASE option is in effect for both SYS001 and SYS002. When you read from the terminal with either SYS001 or SYS002, the input will be translated to uppercase.
5. To obtain a list of current assignments, use the LISTIO command.
6. To cancel all current assignments (that is, to unassign them), you can enter, in succession, the commands:

7. If you want to access VSE private libraries, you must assign the logical units SYSSLB (source statement library), SYSRLB (relocatable library), and SYSCLB (core image library), and you must issue the DLBL command to establish a file definition.
8. An assignment of a logical unit to a disk or directory (specified by *mode* or *DISKm*) should be accompanied by a DLBL command that provides the file identification.
9. If no tape options are specified and 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.
10. 8809 tape drives require the 9TRACK and DEN 1600 options. These are the default options; it is not necessary to state them explicitly.
11. The TRACK and DEN options are not applicable for a 9346 cartridge tape drive and should not be used.

Examples

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

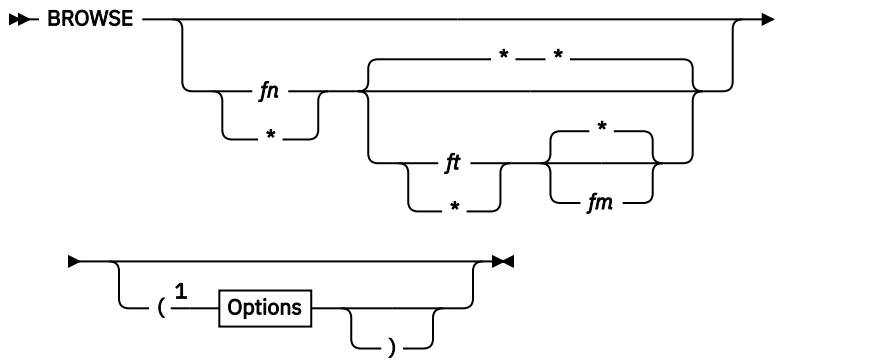
```
assgn sys010 b
```

Messages and Return Codes

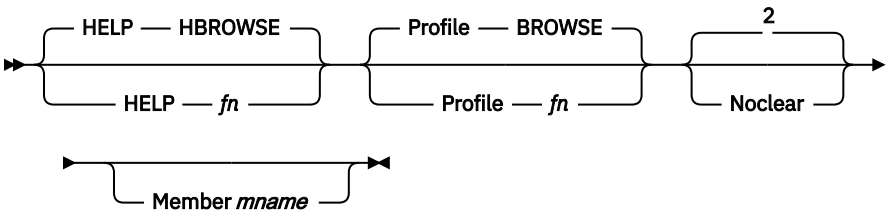
- DMS003E Invalid option: *option* [RC=24]
- DMS027E Invalid device *devtype* [for SYSaaa] [RC=24]
- DMS028E No logical unit specified [RC=24]
- DMS029E Invalid parameter *parameter* in the option *option* field [RC=24]
- DMS050E Parameter missing after SYS*aaa* [RC=24]
- DMS065E *option* option specified twice [RC=24]
- DMS066E *option1* and *option2* are conflicting options [RC=24]
- DMS069E Filemode *mode* not accessed [RC=36]

- DMS070E Invalid parameter *parameter* [RC=24]
- DMS087E Invalid assignment of SYS*aaa* to device *devtype* [RC=24]
- DMS090E Invalid device class *devclass* for *devtype* [RC=36]
- DMS099E CMS/DOS environment not active [RC=40]
- DMS113S Tap*n*(*vdev*) not attached [RC=100]
- DMS115S Device *name* cannot write the *format* recording format [RC=24]
- DMS115S Device *name* cannot write any {9TRACK|compacted} recording formats [RC=24]

BROWSE



Options



Notes:

[1] You can enter Options in any order between the parentheses.
[2] The default is to clear the screen.

Authorization

General User

Purpose

Use the BROWSE command to scan a file or a member of a library. Once in BROWSE, you can use the BROWSE subcommand 'ENTER' to get a split-screen BROWSE of different files or different levels within the same file.

The file may contain character strings that consist of characters from the Single-Byte Character Set (SBCS) and characters from the Double-Byte Character Set (DBCS). The shift-out (SO) and shift-in (SI) control characters identify the DBCS character strings. SO (X'0E') indicates the beginning of the DBCS string and SI (X'0F') indicates the end of the DBCS string.

Operands

fn
is the name of the file to be browsed.

ft
is the type of the file to be browsed.

fm
is the mode of the file to be browsed.

***
is a wildcard to be used if you do not know the file name, file type or file mode. This is the default for file type and file mode.

If you specify only a file name, BROWSE will display the first file it finds with that file name. If you specify BROWSE *, BROWSE will display the first file found on the first disk accessed.

The following are subcommands of browse:

BACKWARD	ETMODE	TOP
BOTTOM	FORWARD	UP
CASE	LEFT	VIEW
DICT	MEMBER	/*string*
DSPF	QUIT	*n*
DOWN/NEXT	RIGHT	=
ENTER	SET	?

Options

You can specify one or more of the following options, separated by blanks or enclosed in parentheses:

Help *fn*
specifies the file name of an exec to be called on invocation of the Help function (PF1). The file name of the exec is not checked for until the HELP function is actually invoked. If this option is not specified, the exec name used by default is HBROWSE.

Profile *fn*
specifies an alternate file name to be used as the BROWSE profile. The file type must be $PROFILE. If *fn* $PROFILE does not exist, or if this option is not specified, the default profile used is BROWSE $PROFILE *.

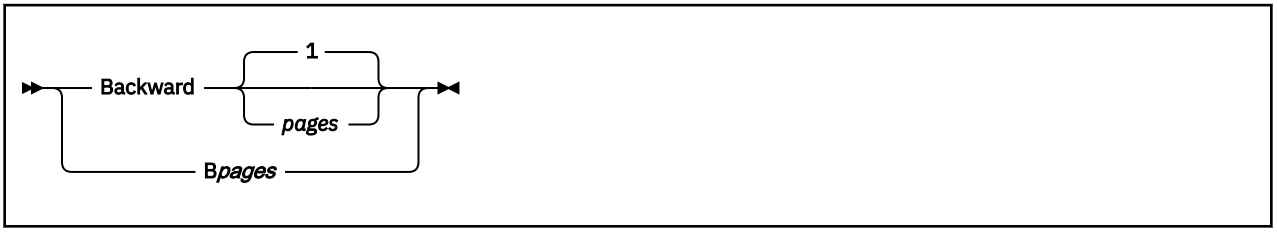
Noclear
specifies not to clear the screen on entry to BROWSE. The default is to clear the screen, then display the first screen of the file to be browsed.

Member *mname*
displays the specified member of the library (such as a MACLIB) immediately on entry to BROWSE.

BROWSE Subcommands

Subcommands are available to help you browse a file or library member. Unrecognized or invalid subcommands will either produce an error message or will be left in the input area and the audible alarm will be sounded.

BACKWARD Subcommand



The BACKWARD subcommand scrolls the screen window backward (toward the top of the file).

pages
is the number of pages to scroll backward. (A page consists of the data lines displayed below the level command line.) If you specify a number greater (or smaller) than the number of valid pages, the alarm will sound for an invalid command. The command will remain on the command line. The default is one page.

BOTTOM Subcommand

The BOTTOM subcommand displays the last page of a file or library member.

CASE Subcommand

The CASE subcommand sets the case used by BROWSE's LOCATE (/) subcommand, which is described later in this section. CASE controls whether the LOCATE subcommand is sensitive to uppercase or lowercase characters. If you just enter `Case`, BROWSE displays the current setting (U or M).

U
specifies uppercase characters. The string and all data from the file is translated to uppercase before the comparison is done during later LOCATE commands. This means that case is effectively ignored for the comparison. This is the initial setting.

M
specifies mixed uppercase and lowercase characters. No translation is done on the string or the file data before the comparison is done in later LOCATE commands, so the string must be an exact match with file data to be found.

If ETMODE is ON, the setting of the CASE subcommand does not affect DBCS characters in valid DBCS strings.

DICT Subcommand

The DICT subcommand displays the names of the members in the library being browsed. This subcommand is only valid for libraries with fixed format, 80-character records.

DSPF Subcommand

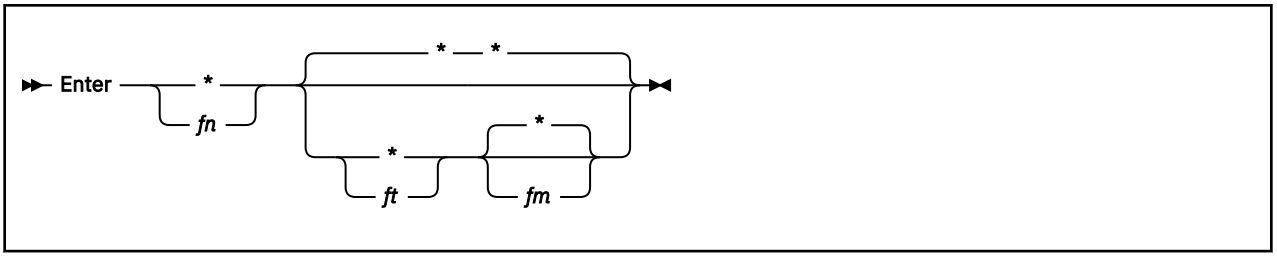
The DSPF subcommand displays the current settings of the PF keys for BROWSE. Press Enter to return to the file you were browsing.

DOWN/NEXT Subcommand

The DOWN/NEXT subcommand moves the screen window down the indicated number of lines (toward the end of the file). If you specify a number greater (or smaller) than the number of valid lines, the alarm will sound for an invalid command. The command will remain on the command line.

lines
specifies the number of lines to move. The default is one line.

ENTER Subcommand

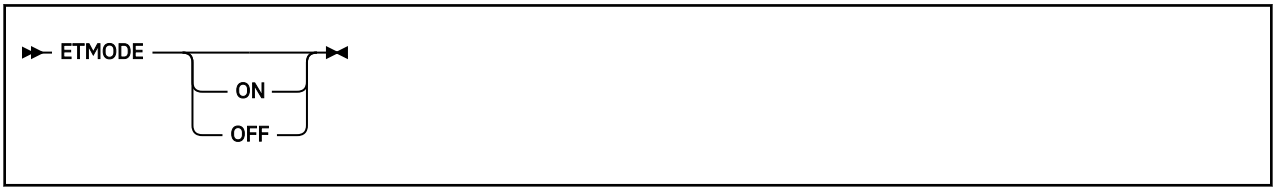


The ENTER subcommand creates a new BROWSE level within the current file or a split-screen display of a file not currently displayed.

fn ft fm
is the file name, file type, and file mode of the new file to be browsed.

*****
indicates the corresponding part of the file ID from the level where the subcommand was entered. However, if the file is not found, the '*' is treated as a wildcard. After substitution, all levels are checked to see if that file is already active. If so, it is displayed. If the file is not yet active, a new level is created that replaces the level from which the subcommand was entered.

ETMODE Subcommand



The Extended Text Mode (ETMODE) subcommand controls the display of DBCS strings. The initial setting of ETMODE depends on whether the console can display double-byte characters. If the console has double-byte capability, ETMODE is set to ON upon entry into BROWSE; if not, ETMODE is set to OFF.

ON
accepts DBCS characters for display (if the console is capable of DBCS display); ignores DBCS characters if CASE is set to U; accepts DBCS characters during the execution of the LOCATE (/) subcommand.

OFF
accepts all characters as SBCS characters and displays the SO and SI control characters as a double quotation mark (").

If you enter the ETMODE ON subcommand at a console that does not support DBCS, the SO and SI characters are displayed as a double quotation mark (") and all characters are treated as single-byte characters for display. The ETMODE subcommand determines how the LOCATE subcommand treats characters during a search in spite of the console's display capability. See the LOCATE subcommand for details.

ETMODE without any operands displays the current ETMODE setting.

FORWARD Subcommand

The FORWARD subcommand scrolls the screen window forward (toward the end of the file).

pages
is the number of pages to scroll forward. (A page consists of the data lines displayed below the level command line.) If you specify a number greater (or smaller) than the number of valid pages, the alarm

will sound for an invalid command. The command will remain on the command line. The default is one page.

LEFT Subcommand

The LEFT subcommand scrolls the screen window toward the left.

cols
is the number of columns to scroll left. If you specify a number greater (or smaller) than the number of valid columns, the alarm will sound for an invalid command. The command will remain on the command line. The default is the width of the screen.

MEMBER Subcommand

The MEMBER subcommand displays a member within a library.

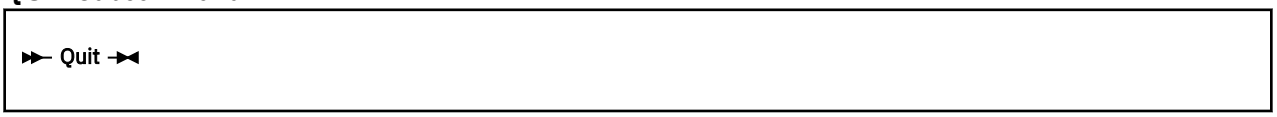
mname
specifies the name of the member to be displayed. This subcommand is only valid for libraries with fixed format, 80-character records.

If no member name is specified, the last selected member is redisplayed. If a member is found, member select mode is entered. (The member name is displayed after the file name in the level header.) All scroll subcommands and the LOCATE (/) subcommand are applied to the selected member only.

To exit from member select mode, use the DICT or QUIT subcommand, or the End PF key.

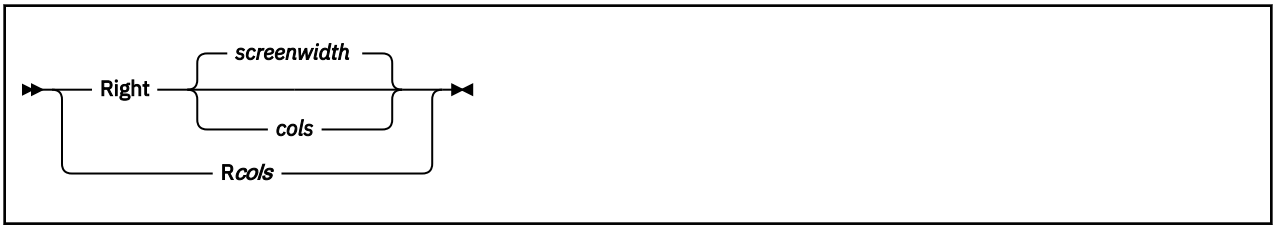
If member select mode was entered through the MEMBER subcommand, the End PF key or the QUIT command will display the dictionary. If member select mode was entered through the MEMBER option, the level is terminated.

QUIT Subcommand



The QUIT subcommand ends the current level (or displays the dictionary if member select mode was entered through the MEMBER subcommand).

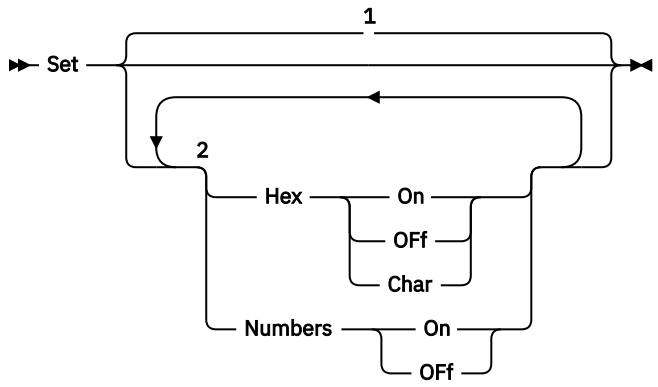
RIGHT Subcommand



The RIGHT subcommand scrolls the screen window toward the right.

cols
specifies the number of columns to scroll right. If you specify a number greater (or smaller) than the number of valid columns, the alarm will sound for an invalid command. The command will remain on the command line. The default is the width of the screen.

SET Subcommand



Notes:

[1] If no parameters are specified, the current settings are displayed.
[2] If multiple options for Set Hex and/or Set Numbers are entered on the command, the last option(s) specified will override the previous options.

The SET subcommand controls display of line numbers and screen data format.

Hex On
Hex OFf
Hex Char
turns hexadecimal (hex) format ON or OFF. ON displays the file in hex format. OFF displays the file in EBCDIC format. CHAR displays the file in both EBCDIC and hex format. The initial setting is OFf.

If ETMODE is ON and the console can display DBCS, any double-byte character in a valid DBCS string is displayed above its corresponding 2 bytes of hex numbers.

If the EBCDIC line of the Hex Char format contains DBCS strings, BROWSE may insert an SO or SI control character to prevent left or right screen border truncation. However, the hex code line or the original data in the string is displayed instead of the inserted X'0E' (SO) or X'0F' (SI) control characters.

Numbers On
Numbers OFf
controls the display of line numbers on the right side of the screen. The initial setting is OFf.

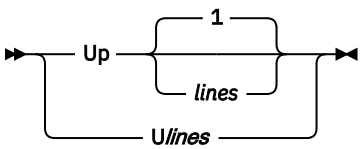
If ETMODE is on, BROWSE still displays line numbers by adjusting the right screen border.

If you do not specify any parameters, the current settings are displayed.

TOP Subcommand

The TOP subcommand displays the first screen of a file or of a library member.

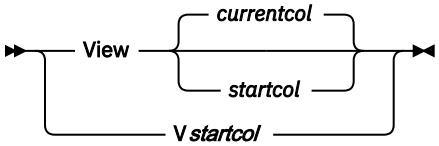
UP Subcommand



The UP subcommand moves the screen window up the indicated number of lines (toward the top of the file).

lines
specifies the number of lines to move the screen up. If you specify a number greater (or smaller) than the number of valid lines, the alarm will sound for an invalid command. The command will remain on the command line. The default is 1.

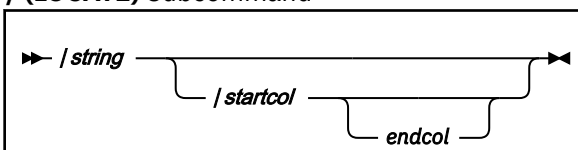
VIEW Subcommand



The VIEW subcommand displays the specified column of the input records in column 1 of the screen.

startcol
specifies the number of the column to be displayed in column 1 of the screen. If no parameter is specified, the current column setting is displayed.

/ (LOCATE) Subcommand



The LOCATE subcommand displays the record containing the specified character string on the level's current line.

The string is translated (or not translated) on the basis of the most recent CASE subcommand setting. If CASE Upper (the default) is in effect, the string (and all data from the file) is translated to uppercase before the comparison is done. If CASE Mixed is in effect, no translation is done, and the string must match the data exactly.

The BROWSE command line is enabled for double-byte character input if the terminal can accept double-byte characters. A user entry of double-byte characters is valid only as the *string* of the LOCATE subcommand. If you enter a double-byte *string* by itself or with any other subcommand, it will result in an UNKNOWN COMMAND message followed by the line entered; this may not be the exact duplicate of the double-byte *string* that was entered.

If ETMODE is ON, translation to uppercase does not take place for valid DBCS characters on either the string or data from the file.

If ETMODE is OFF and CASE is Upper, all characters are treated as SBCS and are translated to uppercase.

string
specifies the character string to be located.

/startcol endcol
specifies column boundaries to be used for the locate operation. If only one column is specified, the search looks for the string beginning in that column only. If beginning and ending columns are

specified, the search looks for the string anywhere between those columns. The closing delimiter is required only if boundaries are given.

When ETMODE is ON:

- DBCS characters in the *string* are compared to the DBCS characters in the file. BROWSE ignores contiguous SBCS characters with the same bit pattern as a DBCS character in the *string*.
- If the *string* is SBCS only, any DBCS strings in the searched file are ignored.
- If the *string* is DBCS only, any SBCS characters in the file are ignored.
- If the *string* contains DBCS characters, the SO or the SI is ignored in the *string* in the following cases:
  - The first character of a *string* is an SO.
  - The last character of a *string* is an SI.
- The column boundary specifications perform alike for all cases, DBCS, SBCS, or mixed DBCS.

When ETMODE is OFF, the search is performed in byte mode and all characters are treated as single-byte characters.

recnum Subcommand

The *recnum* subcommand displays the indicated record number as the level's current line. For example, to display the 100th line, enter 100. If *recnum* is greater than the number of lines in the file, an error message will be displayed.

= Subcommand

The = subcommand repeats the last command for this level. The last command entered for the level will be executed again.

? Subcommand

The ? subcommand displays the last command entered for this level in the level command area. This command will be executed only if the user presses the Enter key.

If two question marks are entered, the next to last command entered for this level is displayed. If more than two ??'s are entered, only the first two are considered and all subsequent input is ignored.

Other Functions Available in BROWSE

& Prefix Function: The & character can be used as a prefix for BROWSE subcommands to leave the subcommand in the display area after execution (for example, `&/string`). This allows repetitive execution.

Program Access Keys: You can press the PA keys while you are in BROWSE:

PA1
enters CP mode. To return to BROWSE, enter:

```
CP BEGIN
```

Note: PA1 will not bring you to CP unless the terminal break key is set to PA1; PA1 is the default setting upon logon. If the terminal break key is not set to PA1, the PA1 key will enter CMS Subset

mode just like the PA2 key. For information about setting the terminal break key, see the CP TERMINAL BRKkey command in *z/VM: CP Commands and Utilities Reference*.

PA2
enters CMS Subset mode. To return to BROWSE, enter:

```
return
```

BROWSE PF Key Settings: Several PF keys are already set for use with BROWSE. The settings and associated BROWSE subcommands are:

PF Key	Subcommand	Meaning
1		Obtains information on the use of BROWSE.
2		Splits screen at cursor position.
3	Quit	Ends the level that contains the cursor.
4		Displays the current cursor position.
5		Repeats last LOCATE (/) command.
6		Line containing cursor becomes current line.
7	Backward	Scrolls backward one screen.
8	Forward	Scrolls forward one screen.
9	Top	Displays first screen in file.
10	Left	Scrolls screen left.
11	Right	Scrolls screen right.
12	BOttom	Displays last screen in file.

Using a BROWSE Profile

At initialization a search is made for BROWSE $PROFILE *, to set PF key functions and to specify special options. The profile may be of fixed or variable format with a logical record length (LRECL) of up to 132. All keywords must be in upper case.

Records recognized:

*OPTION *option*

EXITCLR
forces a clear of the screen on exit from BROWSE. This is useful to prevent confusion when using other full-screen programs from which BROWSE is invoked. The Display Editing System, for example, will abend (end unsuccessfully) if the Clear key is not pressed before pressing Enter, after BROWSE is invoked.

ETMODE ON|OFF
controls the display of DBCS strings. The initial setting of ETMODE depends on whether the console can display double-byte characters. This option will override the automatic setting of ETMODE.

*PFKEYS *n keyword*

n
specifies the number of the PF key to be set.

keyword
defines the function to be assigned to the PF key. The function may be an internal function or a user function. (Only one PF key can be set per record.)

The valid internal functions recognized in *PFKEYS records follow.

Note: Keywords marked with an asterisk are *only* valid when issued from a PF key.

Keyword	Meaning
BOT	Displays the last page of a file or a member.
* CAN	Cancels all levels.
* CCL	Makes the line indicated by the cursor the current line.
* DSPC	Displays cursor position (record number and column).
* END	Ends level at cursor position.
* HELP	Invokes the HELP function.
* RPF	Repeats the last LOCATE command.
* SCB	Scrolls backward (toward top of file).
* SCF	Scrolls forward (toward end of file).
* SCL	Scrolls left the screen width.
* SCR	Scrolls right the screen width.
* SPL	Moves the split to the line indicated by the cursor.
TOP	Displays first page of a file or a member.

The following is a sample BROWSE profile:

```
************************************************************************
**                                                                    **
**                     BROWSE Profile                                 **
**                                                                    **
************************************************************************
*PFKEYS 01 HELP Invoke HELP function
*PFKEYS 02 SPL  Split screen at cursor position.
*PFKEYS 03 END  End the level the cursor is pointing at.
*PFKEYS 04 DSPC Display cursor position.
*PFKEYS 05 RPF  Repeat last find command.
*PFKEYS 06 CCL  Make line at cursor level current line.
*PFKEYS 07 SCB  Scroll backward (to top of file).
*PFKEYS 08 SCF  Scroll forward (to end of file).
*PFKEYS 09 TOP  Display first screen.
*PFKEYS 10 SCL  Scroll display left.
*PFKEYS 11 SCR  Scroll display right.
*PFKEYS 12 BOT  Display last screen.
*PFKEYS 13 HELP Invoke HELP function
*PFKEYS 14 SPL  Split screen at cursor position.
*PFKEYS 15 END  End the level the cursor is pointing at.
*PFKEYS 16 DSPC Display cursor position.
*PFKEYS 17 RPF  Repeat last find command.
*PFKEYS 18 CCL  Make line at cursor level current line.
*PFKEYS 19 SCB  Scroll backward (to top of file).
*PFKEYS 20 SCF  Scroll forward (to end of file).
*PFKEYS 21 TOP  Display first screen.
*PFKEYS 22 SCL  Scroll display left.
*PFKEYS 23 SCR  Scroll display right.
*PFKEYS 24 BOT  Display last screen.
```

Usage Notes

1. BROWSE loads itself as a nucleus extension.
2. When ETMODE is ON, a valid DBCS string is adjusted to prevent the left screen border or the right screen border from truncating a DBCS string.
3. DBCS input is enabled if the console is DBCS capable and the console supports the Input Control extended field attribute. BROWSE adjusts the 3270 data stream attributes to allow the user to shift into double-byte mode at the command line.
4. Each DBCS string is checked for a beginning SO, an ending SI, and an even number of bytes between the SO and SI.

If the string has these three features, the string is treated as a valid DBCS string. If any of these features is missing, the SO or SI is changed to a double quotation mark (") and all characters in the string are treated as single-byte characters.

5. The double-byte characters within a valid DBCS string are checked for a blank (X'4040'), a null (X'0000'), or the first and second byte in the range X'41' to X'FE'.

   If the double-byte character is not valid, it is changed to the double-byte NONDISPLAY character (X'427F'). However, the double-byte string remains valid.
6. BROWSE is not a reentrant module.
7. Extra characters or parameters entered with subcommands may be ignored.
8. BROWSE supports the following screen sizes: 24 x 80, 32 x 80, 43 x 80. Screen sizes other than these are forced to 24 x 80. For example, screen size 27 x 132 is forced to a 24 x 80 screen size.

Messages and Return Codes

- DMS069E Filemode *mode* not accessed [RC=36]
- DMS104S Error *nn* reading file *fn ft fm* from disk or directory [RC=1*xxx*]
- DMS109S Virtual storage capacity exceeded [RC=2|4*xxx*]
- DMS514E Return code *nn* from *command* [RC=1*xxx*]
- DMS618E NUCEXT failed [RC=5*xxx*]
- DMS639E Error in *routine* routine; return code was *nnnn* [RC=*xx*]
- DMS1184E File not found or you are not authorized for it [RC=28]
- DMS2154E File *fileid* is migrated and implicit RECALL is set to OFF [RC=50]
- DMS2155E DFSMS/VM error occurred during creation or recall of file *pathname* [RC=51]
- DMS2156E You cannot BROWSE a file while in subset if the subset environment was entered through BROWSE. [RC=200]
- DMS2189E DMSRLD failed with return code *rc* [RC=6*xxx*]
- DMS2190E Invalid console type or console disconnected. [RC=1]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

Return codes:

RC
Meaning

0
Command complete—no errors

1
Invalid console type (or disconnected)

2
Insufficient free storage for execution

24
Error from CMSCALL PLIST

28
File not found or not authorized to view file

30
Failure to obtain buffer storage

34
Error querying user console

36
File mode not accessed

50
File missing

51
Error reading file

xx
Error from PARSECMD

100
Explanation complete (when ? specified)

200
BROWSE recursion error

1xxx
where xxx is RC from DMSERDRD or FSSTATE

4xxx
xxx RC from DMSRLD

where:

4032
Storage not available for header record buffer

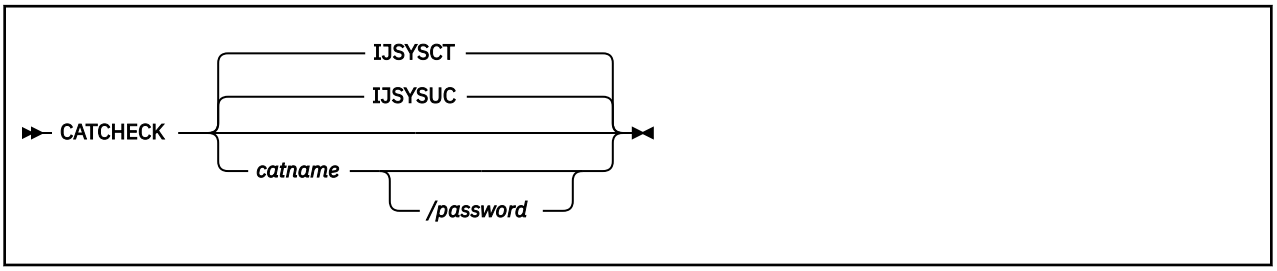
4033
Storage not available for module

4034
Storage not available for RLD buffer

5xxx
xxx is RC from NUCEXT

6xxx
xxx is RC from DMSRLD

CATCHECK



Authorization

General User

Purpose

Use the CATCHECK command to invoke the VSE/VSAM Catalog Check Service Aid to verify a complete catalog structure. This produces a print file containing the catalog analysis. A CMS VSAM user (with or without DOS set ON), can use the CATCHECK command.

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. If you do not specify a catalog name with CATCHECK, it uses the default catalog specified with the DLBL command.

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 by 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 by 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. For more information on how an assembler language program can supply an extended plist, see *z/VM: CMS Macros and Functions Reference*.

Examples

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

```
catcheck private.cat1
```

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

Messages and Return Codes

- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS803E Invalid parameter specification [RC=4]
- DMS804S Error establishing CMS/DOS environment [RC=8]
- DMS805S Error assigning output to printer [RC=12]
- DMS806S VSE/VSAM phase IKQVCHK not found [RC=16]
- DMS807S Error encountered issuing ASSGN for catalog [RC=20]

CLRSCRN

Authorization

General User

Purpose

Use the CLRSCRN command to clear the screen and ensure a clean display.

Operands

MORE
is an optional parameter that forces VM "MORE..." status. This puts you (or an exec user) in control, so you can clear or hold the screen at your discretion. This operand has no effect while in a CMS FULLSCREEN session and will be ignored if used.

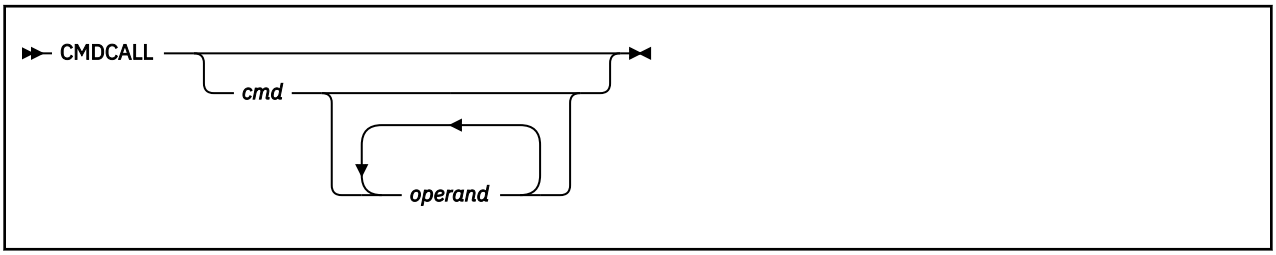
Usage Notes

When issued with no parameters, CLRSCRN immediately clears the console screen.

Messages and Return Codes

- DMS070E Invalid parameter *parameter* [RC=8]
- DMS2157E Error opening the console; CONSOLE return code = *rc*

CMDCALL



Authorization

General User

Purpose

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

Operands

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

operand
is the operand to be passed with the command. You can specify any number of operands.

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 by 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 by the CMDCALL function.

Examples

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

Authorization

General User

Purpose

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.

Operands

sysname
is the eight-character identification of the saved system specifically generated for CMS batch operations with the CP SAVESYS command. For more information on the CP SAVESYS command, see *z/VM: CP Commands and Utilities Reference*.

Note: If *sysname* is not supplied on the command line, 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 more information on how to send jobs to the CMS batch virtual machine, see *z/VM: CMS Application Development Guide*. For more information on setting up a batch virtual machine, see *z/VM: CP Planning and Administration*.
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 through the real card reader. For more information, see *z/VM: CMS Application Development Guide*.
5. If the CMSBATCH command encounters recursive abends, the message "CMSBATCH system ABEND" appears on the system operator's console.

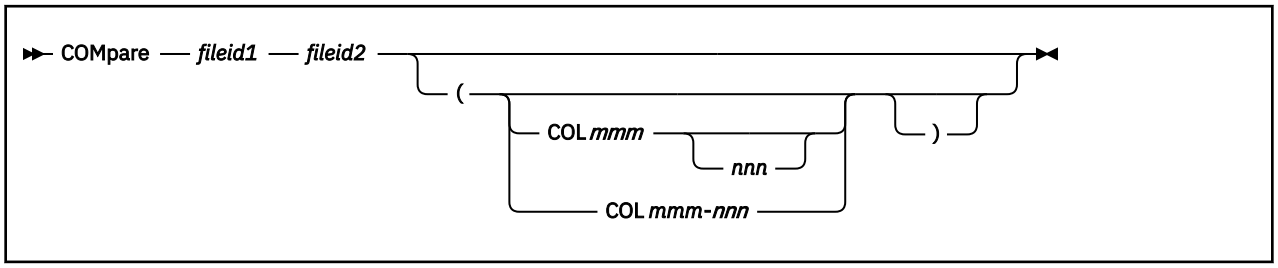
Messages and Return Codes

- DMS100E No batch processor available [RC=40]
- DMS101E Batch not loaded [RC=31 | 55 | 70 | 76 | 88 | 99]
- DMS105E No job card provided
- DMS106E /JOB card format invalid
- DMS107E CP/CMS command *command* [,*devtype*] not allowed [RC=100]
- DMS108E /SET card format invalid [RC=88]
- DMS109E {CPU | Printer | Punch} limit exceeded

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in loading a file	“Messages and Return Codes” on page 485

COMPARE



Authorization

General User

Purpose

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

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 the file identifier in that position is identical with 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).

Options

COL
is a keyword that begins the definition of specific columns to be compared.

mmm
nnn
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.
4. The display of dissimilar records is restricted to a length of 130.

Responses

```
DMS179I  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

- DMS003E Invalid option: *option* [RC=24]
- DMS005E No COLUMN specified [RC=24]
- DMS009E Column *col* exceeds record length [RC=24]
- DMS010E Premature EOF on file *fn ft* [*fm*] [RC=40]
- DMS011E Conflicting file formats [RC=32]
- DMS019E Identical fileids [RC=24]
- DMS029E Invalid parameter *parameter* in the COLUMN field [RC=24]
- DMS054E Incomplete fileid specified [RC=24]
- DMS062E Invalid * in fileid [RC=20]
- DMS104S Error *nn* reading file *fn ft fm* from disk or directory [RC=100]
- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS209W Files do not compare [RC=4]
- DMS211E Column fields out of sequence [RC=24]
- DMS1144E Implicit rollback occurred for work unit *workunitid* [RC=31]
- DMS1252T Rollback unsuccessful for file pool *filepoolid*

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411
Errors in the Shared File System	“File Pool Server Messages” on page 1414
Errors in using a file	“File Error Messages” on page 1418

CONWAIT

Authorization

General User

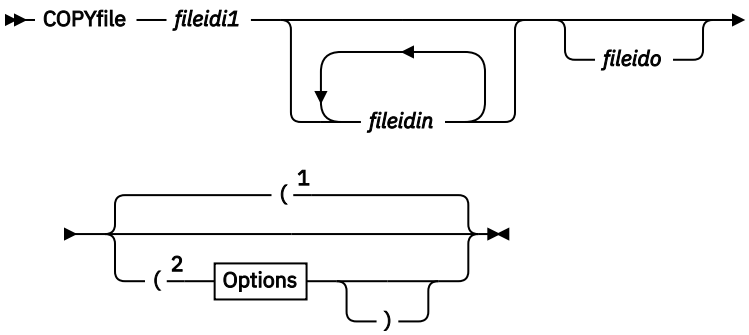
Purpose

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

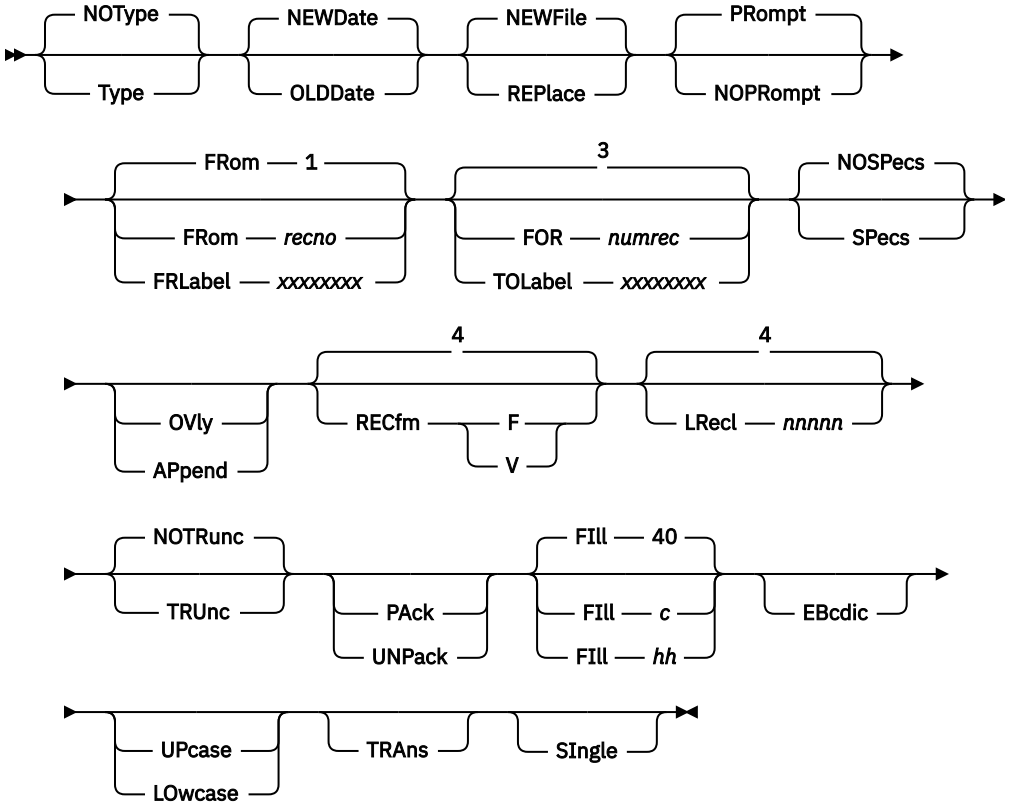
Usage Notes

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

COPYFILE



Options



Notes:

[1] The defaults you receive appear above the line in the Options fragment.
[2] You can enter Options in any order between the parentheses.
[3] The default is for all records to be copied.
[4] The default is the same as the input file.

Authorization

General User

Purpose

Use the COPYFILE command to copy and modify files on CMS minidisks or in Shared File System (SFS) directories. You can also copy files between minidisks and directories. You can:

- Combine two or more files into a single file.
- Copy multiple input files into multiple output files.
- Change file characteristics (such as file mode number and record format) and/or modify file contents.

Operands

fileidi1
is the first (or only) input file. Each file ID component (file name, file type, and file mode) must be specified either by specific name, or by coding an asterisk.

fileidin
are additional input files that may be specified. Each file ID component (file name, file type, and file mode) must be specified. In single output mode, any of the three input file ID components may be specified either by naming the specific component or by coding an asterisk. However, all three file ID components of any additional file ID cannot be specified by asterisks. When creating multiple files, an asterisk is not a valid file ID component. An equal sign (=) may be coded for any of the file ID components, indicating it is the same as the corresponding component in the first file ID specified.

fileido
is the output file(s) to be created. Each component of the file ID (file name, file type, and file mode) must be specified. To create multiple output files, an equal sign (=) must be coded for one or more of the file ID components. If there is only one input file, *fileido* may be omitted, in which case it defaults to = = = (the input file represented by *fileidi* is replaced).

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. This is the default.

NEWDate
uses the current date and time as the date and time of last update of the new file(s). This is the default.

Note: This is not the file creation date. The date of creation for new SFS files is the current date and time, and the date of creation is not changed for replacements of existing SFS files.

OLDDate
uses the date and time on each input file as the date and time of last update of each corresponding output file.

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 existing files are not inadvertently destroyed.

REPlace
causes the output file to replace an existing file with the same file identifier. The default option is REPLACE 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. This is the default.

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

Copy Extent Options

FRom *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 nonblank uppercase characters may be specified.

FOR *numrec*
is the number of records to be copied from each input file. The default for this option is to copy all records.

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 nonblank uppercase characters may be specified.

SPecs
indicates you are going to enter a specification list to define how records should be copied. For more information on how you can define output records in a specification list, see Usage Note "10" on page 88.

NOSPecs
indicates no specification list is to be entered. This is the default.

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

These options can be used to change the record format of a file. For more information, see Usage Note "9" on page 87.

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 *nnnnn*
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 *nnnnn* 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. This is the default.

PAck
compresses records in a file so 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 will be 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.

FIll *c*
FIll *hh*
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 (*hh*) 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). These conversions are made:

From	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 you are going to enter a list of character translations to be made as the file is copied. For more information on entering a list of characters to be translated, see Usage Note "11" on page 89.

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

Incompatible Options

Table 6 on page 82 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 specify any of the options in the second column.

Table 6. COPYFILE Option Incompatibilities

Option	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

Table 6. COPYFILE Option Incompatibilities (continued)

Option	Incompatible Options
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

Usage Notes

1. Two simple uses of the COPYFILE command are:
   - To copy a single CMS file from one minidisk to another, from one SFS directory to another, or between minidisks and directories
   - 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.
2. For those portions of the file identifier you want to stay the same, you may code an equal sign in the output file ID. Thus, the command line just shown 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.
3. 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
```

If you do not specify an output file mode number, it is determined as if the output file:

- Does not exist, the input file mode number is used.
- Exists, the existing output file mode number is used.

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 that of all aliases.

4. 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. Because 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 DMS157E 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.
5. COPYFILE uses a temporary work file called COPYFILE CMSUT1. If the copy operation ends abnormally, the work file may remain on your disk. It will disappear the next time you issue COPYFILE.
6. One of the ways COPYFILE can be used on mixed case file IDs is when FILELIST is entered with the MIXEDON option.
7. Copying Files in SFS Directories

   - To copy files in SFS directories, the directories must first be accessed. For more information, see “ACCESS” on page 30. New files created by using the COPYFILE command have none of the original file's authorities or aliases.
   - When you create a new file in a FILECONTROL directory using COPYFILE, you can create the file even if you have the directory accessed in read-only status, provided you have the proper authority to create the file. If you choose the COPYFILE command to respect the read-only status and not allow you to create the file, use the SET RORESPECT ON command. Upon creating the new file, any users who have NEWREAD or NEWWRITE authority on the target directory will automatically be granted read or write authority, as appropriate.

     For DIRCONTROL directories, you must have the proper authorization and you must have the directory accessed in read-write status to create a new file. Users with DIRREAD authority can read any files that exist in the directory (including the newly-created file), while users with DIRWRITE authority can write to any existing file in the directory.

     For more information on NEWREAD, NEWWRITE, DIRREAD, and DIRWRITE, see “GRANT AUTHORITY” on page 375.
   - You can use the COPYFILE command with the REPLACE option to replace any file for which you have write authority, as well as files in directory control directories for which you have DIRWRITE authority. When the file resides in a FILECONTROL directory, you can replace the file even if you have the directory accessed in read-only status. If you choose the COPYFILE command to respect the read-only status, use the SET RORESPECT ON command. When the file resides in a DIRCONTROL directory, however, you must access the directory in read/write status. In all cases, you must have the proper authorization.
   - When you replace a file, authorities and aliases that existed for the file before 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 example just shown, if PRINT ASSEMBLE is an alias for a base file named PD77MSHC ASSEMBLE, the contents of PD77MSHC ASSEMBLE are replaced.

- When you create a file in another user's directory, 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*) you have locked SHARE, or to a file or directory another user has locked SHARE, UPDATE, or EXCLUSIVE. You cannot copy a file (*fileid1*) locked EXCLUSIVE by another user.
- When using the NEWFILE option, if one or more output files already exist, error message DMS024E is displayed and the COPYFILE command terminates. This option is the default so existing files are not inadvertently destroyed.
- When using COPYFILE, the recoverability and overwrite attributes (also called extended file attributes) of the output file depend on whether the output file is a new file or a replacement of an existing file. If you are copying to a file that does not already exist, the extended file attributes of the newly created output file will be set to the current system defaults. If you are replacing a file with the COPYFILE command, the extended file attributes of the existing output file will be unchanged.

  Note: The extended file attributes of a file copied into a downleveled server will be set to RECOVER and NOTINPLACE.

Empty Files

If you specify OVLY and the overlay file is empty, or if you specify FROM, FOR, FRLABEL, TOLABEL, PACK, or UNPACK and the source file is empty, no file is created and you get error message DMSCPY1229E.

If all input files are empty, and you are copying into a minidisk, or an earlier directory in a file pool that does not support empty files, no file is created. Even if the output file already exists and you have specified REPLACE, the file is not replaced. Error message DMS173E is issued with the following text:

```
Empty output file fn ft fm not created
```

If you are copying empty files into a directory in a file pool server with empty file support, an empty file will be created. If the file already existed and you specify REPLACE, the file will be replaced with an empty file.

SFS Migrated Files

If DFSMS/VM is being used to manage your file pool repositories, you may have files that appear to reside in your file pool, but whose data DFSMS/VM has migrated to its storage repository. If SET RECALL is ON, this data is automatically recalled when referenced. If SET RECALL is OFF, you will get message DMSCPY2154E if you try to use a migrated file as an input file to COPYFILE (or an output file if you use the APPEND option.)

External Objects

COPYFILE does not copy external objects. Error message DMS002 is issued with the following text:

```
Input file(s) fn ft fm not found
```

8. 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. TEST DATA3 will consist of TEST DATA2 appended to TEST DATA1.

Note: If any input file has a file mode number of 3, it is possible 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.

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.

Using Wildcard Characters

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.

Note: If you code asterisks in an input file ID, remember CMS will also search any disks or directories you have accessed as extensions of file mode A. For example, if you enter

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

not only will all files on the disk or directory accessed as A be copied onto the one accessed as B, but all files on any extensions of file mode A will be copied as well. 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.

Note: If the input file mode is an '*', 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 the file already exists. For example, if you have a file named 'C B A' and you issue the command 'COPY C * * = D =', COPYFILE will first create an output file named 'C D A'. This file will then match the input file ID of the file 'C * *' and COPYFILE will attempt to write an output file with the name 'C D A', which already exists.

Similarly, when copying with a given file mode, do not specify an asterisk for both file name and file type of the input file when NEWFILE is specified or implied. This is because COPYFILE will generate an output file that matches the input file ID, which already exists.

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

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

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

```
copyfile x * a1 = file = (single
```

combines all files with a file name of X on the disk 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 these copy operations you can choose 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 files ALPHA TEST A and ALPHA OUTPUT A were copied into a file named ALPHA SUMMARY A and files BETA TEST A and BETA OUTPUT A were copied into a file named BETA SUMMARY A.

9. 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 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 *
   ```

   each record in the file SHORT RECS is padded with asterisks to the record length. Assuming SHORT RECS was originally a variable-length file, the record length is taken from the longest existing record.

   Note: 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, 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. Because most fixed-length files are blank-padded to the record length, you do not need to specify the FILL option unless you know some other character appears more frequently.

Packed files are fixed format with a logical record length of 1024, with one exception. Previous releases of CMS allowed a disk blocksize of 800 bytes. A file which was packed on an 800-byte blocksize disk and subsequently copied to a 512, 1KB, 2KB, or 4KB blocksize disk will be fixed format with a logical record length of 800. A packed file logical record length 800 can be unpacked back to its original specifications regardless of the blocksize of the disk it resides on, and should be unpacked and re-packed if you need to minimize disk block usage.

For more information on how to convert record formats on packed files with the COPYFILE command you can specify single or multiple output files, see Usage Note "9" on page 87. For example:

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

compresses all assembler files on 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.

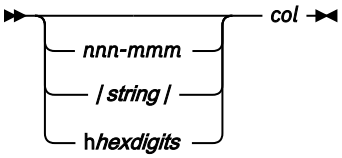
10. Entering a COPYFILE Specification List

    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:

    ```
    DMS601R  Enter specification list:
    ```

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

    Where:

    nnn-mmm
    specifies the start and end columns of the input file 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
    specifies any string of uppercase and lowercase characters or numbers delimited by any non-alphanumeric character (shown as / in the syntax.)

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

hhexdigits
specifies an even number of hexadecimal digits prefixed with an h.

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
```

```
DMS601R  Enter specification list:
```

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

This means *put a vertical bar at column one of the output file, at column three put the first eight characters from the input file; at column 12 of the output file put a vertical bar, a string of three asterisks at column 14, then put the original contents of input file columns 9-80 from column 18 to the remainder of the record in the output file*. After this command is executed, each record in the file SORTED LIST will look like this:

```
| oooooooo | *** oooo....
```

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, if your specs are 1-15 6:

```
copyfile sorted list a (specs noprompt lrecl 20 fill $
```

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 the file type 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.

11. 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:

    ```
    DMS602R  Enter translation list:
    ```

    and a read is presented to your virtual machine. You may enter the translation list. If you do not choose 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
```

```
DMS602R   Enter translation list:
```

```
* - A f0 00 ff
```

specifies 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.

12. Use OPENVM GETBFS and OPENVM PUTBFS to copy SFS and minidisk files between the CMS record file system and the byte file system (BFS).

Responses

```
DMS601R  Enter specification list:
```

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

```
DMS602R  Enter translation list:
```

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

```
DMS721I 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.

```
DMS132S  File fn ft fm too large
```

This message appears when the input file is too large for COPYFILE to handle.

Messages and Return Codes

- DMS002E File[(s)] [*fn* [*ft* [*fm*]]] not found [RC=28]
- DMS002E Input file[(s)] [*fn* [*ft* [*fm*]]] not found [RC=28]
- DMS002E Overlay file[(s)] [*fn* [*ft* [*fm*]]] not found [RC=28]
- DMS003E Invalid option: *option* [RC=24]
- DMS024E File *fn ft fm* already exists; specify REPLACE option [RC=28]
- DMS029E Invalid parameter *parameter* in the option *option* field [RC=24]
- DMS030E File *fn ft fm* already active [RC=28]
- DMS037E Filemode *mode*[(*vdev*)] is accessed as read/only [RC=36]
- DMS037E Base file for *fn ft fm* is in a DIRCONTROL directory accessed read/only [RC=36]
- DMS042E No fileid(s) specified [RC=24]
- DMS048E Invalid filemode *mode* [RC=24]

- DMS054E Incomplete fileid specified [RC=24]
- DMS062E Invalid character *char* in fileid *fn ft* [*fm*] [RC=20]
- DMS063E No [sort|translation|specification] list {entered|given} [RC=40]
- DMS064E Invalid [translate] specification at or near *list* [RC=24]
- DMS065E *option* option specified twice [RC=24]
- DMS066E *option1* and *option2* are conflicting options [RC=24]
- DMS067E Combined input files illegal with PACK or UNPACK options [RC=24]
- DMS068E Input file *fn ft fm* not in packed format [RC=32]
- DMS069E Filemode *mode* not accessed [RC=36]
- DMS101S SPECS temp string storage exhausted at *storarea* [RC=88]
- DMS102S Too many fileids [RC=88]
- DMS103S Number of SPECS exceeds maximum *nn* [RC=88]
- DMS104S Error *nn* reading file *fn ft fm* [RC=100]
- DMS105S Error *nn* writing file *fn ft fm* on disk or directory [RC=100]
- DMS156E FROM *nnn* not found--the file *fn ft fm* has only *nnn* records [RC=32]
- DMS157E Label *label* not found in file *fn ft fm* [RC=32]
- DMS172E TOLABEL *label* {equals|is an initial substring of} FRLABEL *label* [RC=24]
- DMS173E No records were copied to output file *fn ft fm* [RC=40]
- DMS173E Empty output file *fn ft fm* not created [RC=40]
- DMS173E No records were copied to output file *fn ft fm* [RC=40]
- DMS901T Unexpected error at *vstor1*: plist *function fn ft fm* at vstor2, base vstor3, rc nn [RC=256]
- DMS903T Impossible PHASE code *xx* [RC=256]
- DMS904T Unexpected UNPACK error at *vstor1*, base *vstor2* [RC=256]
- DMS516E An existing variable-length record in an SFS file cannot be replaced with one of a different length. [RC=32]
- DMS1141W User filespace threshold [still] exceeded [for file pool *filepoolid*]
- DMS1144E Implicit rollback occurred for work unit *workunitid* [RC=31]
- DMS1184E File *fn ft fm* not found or you are not authorized for it [RC=28]
- DMS1229E OVERLAY|INPUT file *fn ft fm* is empty [RC=32]
- DMS1252T Rollback unsuccessful for file pool *filepoolid*
- DMS1258E You are not authorized to write to file *fn ft fm* [RC=76]
- DMS1259E File pool *filepoolid* has run out of physical space in the storage group [RC=40]
- DMS1262S Error *nnn* closing file *fn ft fm* [RC=31]
- DMS2153E File *fileid1* is migrated and DFSMS/VM Recall processing is not active [RC=51]
- DMS2154E File *fileid1* is migrated and implicit RECALL is set to OFF [RC=50]
- DMS2155E SFS error *errorid* in filepool *filepoolid* occurred during recall of file *fn ft fm* [RC=51]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411
Errors in the Shared File System	“File Pool Server Messages” on page 1414
Errors in using a file	“File Error Messages” on page 1418

CP

Authorization

General User

Purpose

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

Operands

command
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 in the following instances:
   - 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 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 a CP command that is not valid, you receive a return code of -1. In an exec, this return code is +1.

Examples

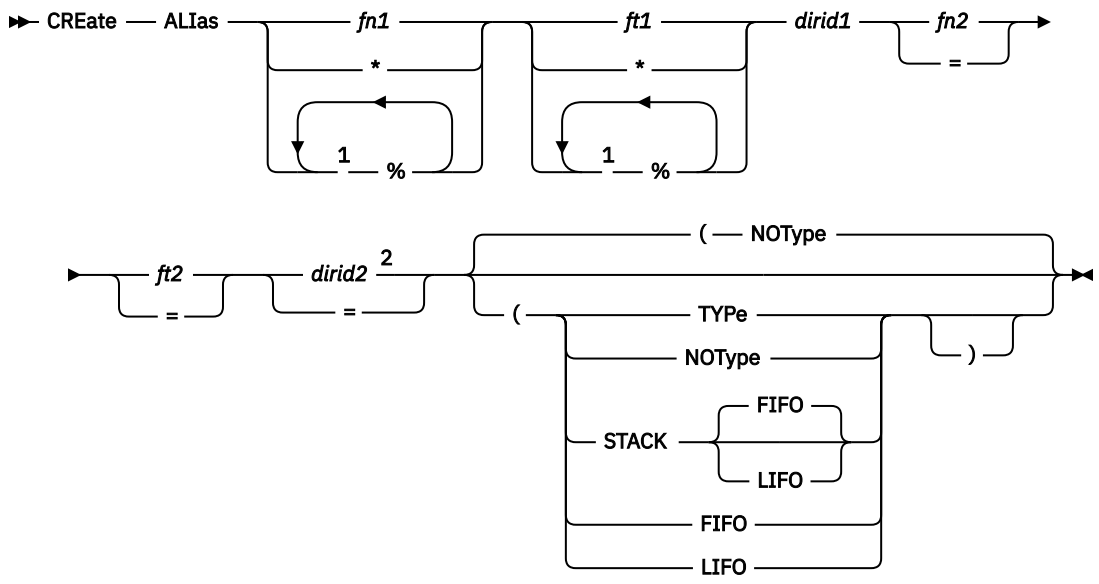
If the implied CP function is set to OFF for your virtual machine (QUERY IMPCP to find out setting), and you want to specify the CP SCREEN command, you can precede it by the CP command. For example:

```
cp screen inarea red
```

Responses

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

CREATE ALIAS



Notes:

[1] Each variable may be composed of up to a combination of 8 characters, including the symbols % and *. If % or * symbols are used, *fn2* or *ft2* must be =.

[2] The variable *dirid2* may only be specified as = if *fn1 ft1* does not equal *fn2 ft2*.

Authorization

General User

Purpose

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 file control directory for which you have write authority. A file control directory is an SFS directory that has the file control (FILECONTROL) attribute.

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.

Operands

fn1 ft1
identifies the file for which you are creating an alias. Special characters (* or %) can be used to designate a set of files, providing you have appropriate authority for the directory specified by *dirid*. For file control directories, you need read or write authority on the directory. For directory control directories, you need DIRREAD or DIRWRITE authority on the directory.

For more information specifying the *dirid*, see "Naming Shared File System (SFS) Directories" on page 6. For information on using special characters to do pattern matching, see "Using Pattern Matching to Specify Sets of Files" on page 14. 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.

For file control directories, only those files for which you have read or write authority will have aliases created. There are no individual file authorizations for directory control directories, so aliases will be created for all files that match the pattern so long as you have either DIRREAD or DIRWRITE authority to the directory.

dirid1
identifies the SFS directory that contains the file(s) on which you are creating the alias. You can also use a file mode for the *dirid* if the directory is already accessed. For more information specifying the *dirid*, see "Naming Shared File System (SFS) Directories" on page 6.

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

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

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 from *fn1 ft1*. To do this, the directory must have the file control attribute and you must have write authority to it.
2. You can specify the name of a file control directory for *dirid2*. You must have write authority on the directory.

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. The default is NOType.

STACK FIFO
STACK LIFO
places the alias name(s) in the console stack rather than displaying it at the terminal. The default is FIFO.

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

LIFO
specifies 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. To create an alias:
   - The target directory must be file control.
   - You must have authority to write to the target directory.
   - You must have explicit authority to the base file (the implicit authority of an administrator is not sufficient).
   - The owner of the target directory must own or have explicit authority to the base file.

   Explicit authority means authority to read or write to the file granted with the CMS GRANT AUTHORITY command or the DMSGRANT CSL routine.
3. If a base file is erased, any alias for that file is changed to an erased alias.

If the explicit authority you have to a base file is revoked, any alias you have for that file is changed to a revoked alias. Authority to a base file can be READ or WRITE to a file in a file control directory, or DIRREAD or DIRWRITE to a directory control directory.

4. Aliases cannot be created in directories that have the directory control (DIRCONTROL) attribute. You can, however, create aliases in file control directories that refer to base files residing in directory control directories.
5. If USER A wants to share your file, you grant USER A authority on the file. For file control directories, you can grant either read or write authority on the file. For directory control directories, you cannot grant read and write authority on individual files. Instead, you can grant directory control read (DIRREAD) or directory control write (DIRWRITE) authority on the directory. Once authority is granted, USER A can then create an alias to that file in any owned file control directory, or in file control directories on which USER A has write authority.

   Another way to share the file is: USER A could grant you write authority on a particular file control directory. You could then create an alias for the file in USER A's directory.
6. Aliases cannot be created for files located in another file pool or for files located on minidisks.
7. An alias cannot be created on a file open by another user.
8. 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.
9. To delete an alias, use the ERASE command or the DISCARD command from FILELIST.
10. You can use the RELOCATE command to move any aliases to another directory.
11. You may create as many aliases as you choose on the base file.
12. Aliases are not created in the specified directory for:

    a. Subdirectories

    b. Erased or revoked aliases

    c. External objects

    d. Files you are not authorized to either read or write

    If special characters are used for *fn* or *ft*, or both, processing continues for the remaining file IDs that match the specified pattern.
13. 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 REXX 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.

For more examples on using the CREATE ALIAS command, see *z/VM: CMS User's Guide*.

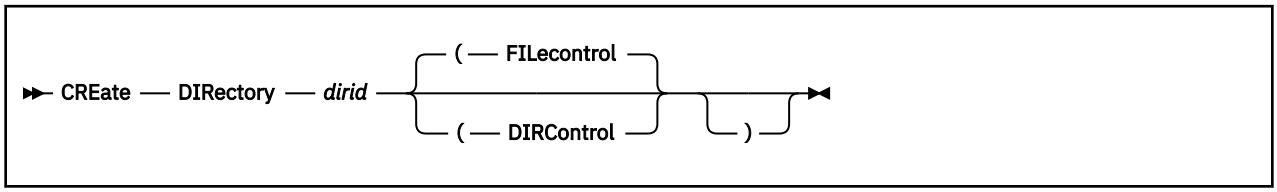
Messages and Return Codes

- DMS002E File *fn ft fm* | *dirname* not found [RC=28]
- DMS069E Filemode *mode* not accessed [RC=36]
- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS1163E The CREATE ALIAS command failed for *fn ft fm* | *dirname* [RC=nn]
- DMS1184E Directory *dirname* not found or you are not authorized for it [RC=28]
- DMS1184E File or directory not found or authorization requirements not met [RC=28]
- DMS1187E Too many subdirectory levels in *dirid* [RC=24]
- DMS1188E Filemode *mode* is not associated with a directory [RC=74]
- DMS1189E Filemode *mode* is associated with a top directory [RC=24]
- DMS1223E There is no default file pool currently defined [RC=40]
- DMS1241E Directories specified are in different file pools [RC=88]
- DMS1312E A filemode number may not be specified with the filemode of an alias [RC=24]
- DMS2040E CREATE ALIAS cannot be performed on a directory control directory [RC=36]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411
Errors in the Shared File System	“File Pool Server Messages” on page 1414

CREATE DIRECTORY



Authorization

General User

Purpose

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.

Operands

dirid
specifies the name of a new directory you are creating. For this command, you cannot use *fm* alone as the *dirid* because this would not name a new directory. For more information specifying the *dirid*, see "Naming Shared File System (SFS) Directories" on page 6.

Options

DIRControl
the DIRCONTROL option establishes the directory control attribute for the directory. Such directories are called *directory control directories* (sometimes *DIRCONTROL directories*). For more information about the characteristics of the directory control directories, see "Usage Notes" on page 97.

FILecontrol
the FILECONTROL option establishes the file control attribute for the directory. Such directories are called *file control directories* (sometimes *FILECONTROL directories*). The default is FILECONTROL.

Unlike directory control directories, file control directories provide control on an individual file basis. For more information, see Usage Note "9" on page 98. In most sharing environments, individual file control is preferred to directory level control.

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 that if you hold an UPDATE or EXCLUSIVE lock on the directory, you can create a subdirectory within the directory.
6. You cannot create a subdirectory in a directory control directory you have accessed read-only or some other user has accessed read/write.
7. 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 REXX exec with ADDRESS COMMAND in effect

8. Some characteristics of file control directories are:
   - All requests for file control directories and the files within them are directed to the file pool server machine. CMS uses Advanced Program-to-Program Communication/VM (APPC/VM) to communicate with the server. The server checks authorization and implicitly locks necessary resources whenever a file is opened.
   - Any number of users can access a file control directory in read/write status. These users can write to different files within the directory at the same time. SFS ensures no two users write to the same file at the same time.

     Any number of users can access file control directories in read-only status. Authorized users can read files concurrently being written to by some other user. SFS ensures the readers see a consistent version of the file from the time it is opened until the time it is closed. Except for files having the INPLACE attribute, readers see committed changes when they close and reopen a file. Readers can see changes to INPLACE files as they are written.
   - Users cannot access the directory unless they have either READ or WRITE authority for that directory.
   - The directory can contain aliases that refer to files in file control or directory control directories.
   - A file in a file control directory can be the source or target of a RELOCATE command.
   - Directories can be relocated from or to a directory with the FILECONTROL attribute, even if the directory is accessed.
   - The directory and its files can be locked UPDATE, SHARE, or EXCLUSIVE.
   - READ and WRITE authority can be granted on file control directories or on individual files within file control directories. NEWREAD and NEWWRITE authority can also be granted on file control directories. See the GRANT AUTHORITY command for more information about these authorities.
   - The creator (owner) of a file control directory has WRITE and NEWWRITE authority by default.
   - You can create a subdirectory in a file control directory accessed in read-only mode.
9. Some characteristics of directory control directories are:
   - Performance benefits are possible with directory control directories. A file pool administrator can make a directory control directory eligible for use in a *data space*. A data space is an area of virtual storage that can be shared by users on the same processor. When you access a directory control directory in read-only status, CMS will read directory and file data directly from the data space. This avoids the overhead of communicating with the file pool server. Also, because the data resides in virtual storage, there is minimal DASD overhead. Moreover, authorization checking and locking occur at access time rather than each time a file is opened.
   - Directory control directories do not allow the same level of concurrency that file control directories do. While read concurrency is the same, write activity is restricted. In particular, when a user has a directory control directory accessed in read/write mode, no other user can write to the directory. If no one has the directory accessed in read/write mode, different users can concurrently write to different files within the directory by using Callable Services Library (CSL) routines in application programs to directly access the files.

     Only one read/write access, using the CMS ACCESS command, is permitted to the directory at a time. More than one read-only access is permitted concurrently. See the CMS ACCESS command for more information.
   - When a directory control directory is in a data space, references to any INPLACE files in the directory will not use the data space, but will go to the server.
   - Those who access the directory for read-only purposes, using the CMS ACCESS command, will see file changes only when they release and re-access the directory. One exception concerns INPLACE

files. Readers can see changes to INPLACE files as soon as they are written without having to close and reopen the file or reaccess the directory.

- Those who access the directory for read/write purposes, using the CMS ACCESS command, will see any changes to files in the directory as they are made.

  When a directory is accessed (read-only) more than once without first releasing it, all nested accesses must be released before committed changes become available to the user (upon next access).
- Users cannot access the directory unless they have either directory control read (DIRREAD) or directory control write (DIRWRITE) authority.
- Directory control directories cannot contain aliases.
- Aliases in directories that do not have the DIRCONTROL attribute may be used to read from or write to base files in a directory control directory. If you try to write to the alias while someone has the parent directory control directory accessed, the write will fail.
- A file in a directory control directory cannot be the source or target of a RELOCATE command.
- Directories can be relocated from or to a directory with the DIRCONTROL attribute. This requires you do not have the containing directory accessed R/O and no other user has it accessed R/W.
- You cannot relocate a directory with the DIRCONTROL attribute if anyone other than you has the directory accessed using the CMS ACCESS command.
- The directory and its files can only be locked UPDATE.
- No authority other than DIRREAD and DIRWRITE authority can be granted or revoked for the directory.
- The creator (owner) of a directory with the DIRCONTROL attribute has directory control write (DIRWRITE) authority by default.
- If you have a directory with the DIRCONTROL attribute accessed read-only, you cannot create a subdirectory therein.

For more information about directory control directories, see *z/VM: CMS User's Guide*.

Examples

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
```

For more examples on using the CREATE DIRECTORY command, see *z/VM: CMS User's Guide*.

Messages and Return Codes

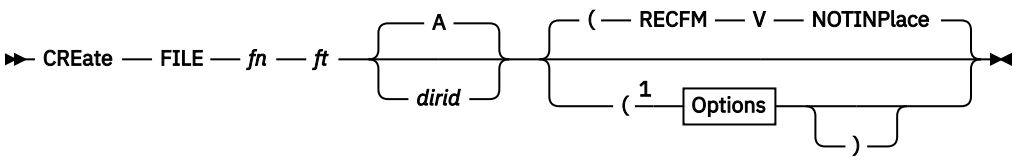
- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS1331E You can not authorized to create a Directory *dirname* [RC=28]
- DMS1184E Directory *dirname* not found or you are not authorized to use CREATE DIRECTORY on this directory [RC=28]
- DMS1187E Too many subdirectory levels in *dirid* [RC=24]
- DMS1188E Filemode *mode* is not associated with a directory [RC=74]
- DMS1189E Filemode *mode* is associated with a top directory [RC=24]
- DMS1218E You cannot create top directories using the CREATE DIRECTORY command [RC=88]
- DMS1223E There is no default file pool currently defined [RC=40]

- DMS2039E DIRCONTROL option is not supported with the current level of file pool *filepoolid* [RC=88]
- DMS2040E CREATE DIRECTORY cannot be performed on a directory control directory that is accessed read-only [RC=36]

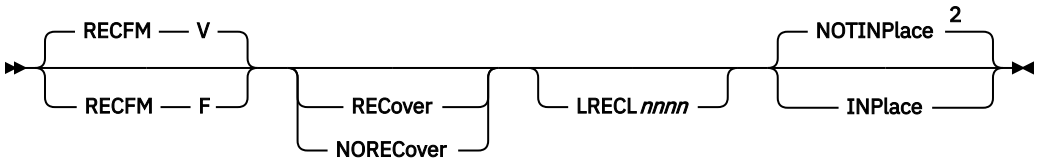
Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411
Errors in the Shared File System	“File Pool Server Messages” on page 1414

CREATE FILE



Options



Notes:

[1] You can enter Options in any order between the parentheses.
[2] The default is NOTINPLACE unless changed by the DMSPUSHA CSL routine.

Authorization

General User

Purpose

Use the CREATE FILE command to create an empty file in an SFS directory.

Operands

fn
is the file name of the file being created.

ft
is the file type of the file being created.

dirid
is the directory identifier of the directory that will contain the file being created. You can also use a file mode for the *dirid* if the directory is already accessed. If not specified, the file will be created in the directory accessed as A. For more information specifying the *dirid* see "Naming Shared File System (SFS) Directories" on page 6.

Options

RECFM
assigns a record format to the file. The valid values are:

F
Specifies fixed length records. You can specify the length in the LRECL option. If you omit LRECL for a fixed length file, length will default to 80 bytes.

V
Specifies variable length records. LRECL of a variable length file is initialized to 1 byte when the file is created, but will change dynamically to the length of the longest record in the file. This is the default.

RECover
indicates an application initiated rollback will cause all uncommitted updates to the file to be discarded.

NORECover
specifies changes to the file are not rolled back, in the event of an application initiated rollback. In most cases, the updates will be committed.

LRECL *nnnn*
defines the logical record length of the file. For more information on this option, see the description of the RECFM option.

INPlace
specifies updates are to be made in place where possible for reduced DASD utilization. Users that have an INPLACE file open for read have to re-open the file to see extensions (new blocks) that have been written and committed to the file. Readers may not see a consistent version of the file from OPEN to CLOSE.

NOTINPlace
specifies the file writes are to be shadowed such that readers see a consistent version of the file from Open to Close, and will not see uncommitted updates.

Usage Notes

1. CREATE FILE creates an empty file. It will have zero records and zero file blocks allocated to it.
2. You must have write authority to the directory to create the file.
3. If you create a file on someone else's directory, both you and the directory owner will have write authority to the file. If the file is in a directory control directory, any users with DIRREAD authority to the directory will have authority to the newly created file for file control directories. If NEWREAD or NEWWRITE authority has been granted on the directory, the grantees will also have authority to the new file.
4. Default values for RECOVER|NORECOVER and INPLACE|NOTINPLACE attributes are determined differently, depending on whether both of them, just one, or neither of them are specified. This matrix shows:

Table 7. File Attribute Setting Matrix for CREATE FILE During File Creation

	NOTINPLACE	**INPLACE**	**(Omitted)**
RECOVER	RECOVER/ NOTINPLACE	error	RECOVER/ NOTINPLACE
NORECOVER	NORECOVER/ NOTINPLACE	NORECOVER/ INPLACE	NORECOVER/ NOTINPLACE
Recoverability Attribute Not Specified	RECOVER/ NOTINPLACE	NORECOVER/ INPLACE	DMSPUSHA (PUSH ATTRIBUTES CSL routine) settings or RECOVER/ NOTINPLACE

5. You can invoke the CREATE FILE command from the command line, from an exec, or as a function from a program. No error messages are issued if CREATE FILE 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 REXX exec with ADDRESS COMMAND in effect

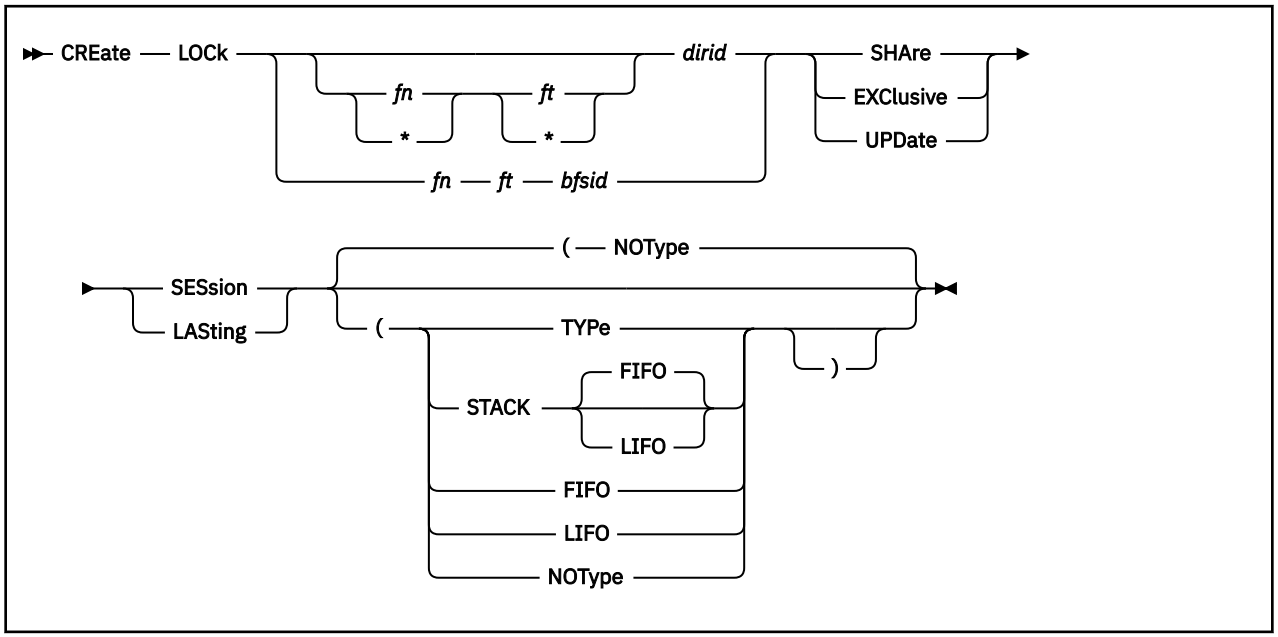
Messages and Return Codes

- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS029E Invalid parameter *length* in the option LRECL field [RC=24]
- DMS065E Option *option* specified twice [RC=24]
- DMS066E *option1* and *option2* are conflicting options [RC=24]
- DMS069E File mode *mode* is not accessed [RC=36]
- DMS1184E Directory *dirname* not found or you are not authorized to use CREATE FILE on this directory [RC=28]
- DMS1187E Too many subdirectory levels in *dirid* [RC=24]
- DMS1188E File mode *mode* is not associated with a directory [RC=74]
- DMS1223E There is no default file pool currently defined [RC=40]
- DMS2040E CREATE FILE cannot be performed on a file in a directory controlled directory that is accessed read-only [RC=36]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411
Errors in the Shared File System	“File Pool Server Messages” on page 1414
Errors in using a file	“File Error Messages” on page 1418

CREATE LOCK



Authorization

General User for SFS files and directories. A file pool administrator may enter this command for byte file system (BFS) files.

Purpose

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

Operands

fn ft
specifies the name of the file to be locked. You must have authority to read from or write to the file to lock it, depending on the type of lock you choose to create. Special characters (* or %) can be used to designate a set of files, providing you have appropriate authority for the directory specified by *dirid*. For file control directories, you need read or write authority on the directory. For directory control directories, you need DIRREAD or DIRWRITE authority on the directory.

For more information specifying the *dirid*, see "Naming Shared File System (SFS) Directories" on page 6. For more information on using special characters to do pattern matching, see "Using Pattern Matching to Specify Sets of Files" on page 14.

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. You can also use a file mode for the *dirid* if the directory is already accessed. For more information specifying the *dirid*, see "Naming Shared File System (SFS) Directories" on page 6.

bfsid
identifies the byte file system (BFS).

For a BFS file, *fn* and *ft* are system-generated values that together uniquely identify the file within the BFS.

SHAre
specifies others may read while you read it. You must have read authority to use the SHARE operand. This operand applies only to directories with the file control attribute and the files that reside within them.

BFSregular files may not be locked SHARE.

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. This operand applies only to directories with the file control attribute and the files that reside within them.

UPDate
specifies others may read while you read or modify. You must have write authority to use this operand on a file control directory or a file within it. You must have directory control write (DIRWRITE) authority to use this operand on a file within a directory control directory.

If UPDATE is specified for a BFS object, it is treated as if EXCLUSIVE was specified.

SESsion
specifies 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
- Re-IPL
- File system server abend
- Network or APPC/VM failure on the last data link with a file pool
- Log off

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

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

STACK FIFO
STACK LIFO
places the output in the console stack rather than displaying it at the terminal. The default is FIFO. Error messages are displayed at the terminal and are not stacked.

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

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

Usage Notes

Applicable to both the shared file system (SFS) and byte file system (BFS)

1. 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. For more information, see the "DELETE LOCK" on page 158.
2. When you use a batch facility that—unlike CMSBATCH—runs the job under your user ID:

- Even if you have an EXCLUSIVE lock on a file, your job running in the batch machine can use the file.
- If your batch machine has a file open to update it, any job running from your own virtual machine cannot update the file.

3. For more information on how the FILEWAIT setting affects the CREATE LOCK command, see the "SET FILEWAIT" on page 933.
4. If the CREATE LOCK command is issued from an exec or assembler program for a file pool that is active on the specified work unit, the command will fail.
5. 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 REXX exec with ADDRESS COMMAND in effect
6. Locking a file or directory, in any lock mode, does not prevent DFSMS/VM from migrating or recalling files.
7. Locking a file or directory EXCLUSIVE or UPDATE prevents anyone other than the holder of the lock from deleting that file, or files, in that directory, while locking SHARE prevents anyone from deleting that file, or files, in that directory.
8. For examples on using the CREATE LOCK command, see *z/VM: CMS User's Guide*.

Applicable to only the shared file system (SFS)

1. Locks are not created in the directory for:
   - Subdirectories
   - Erased or revoked aliases
   - Aliases whose base files reside in directory control directories
   - External objects
   - Files you are not authorized to read or write

   If special characters are used to specify the file name or file type, or both, processing continues for the remaining file IDs that match the specified pattern.
2. When you CREATE LOCK on a alias, it is the base file that determines whether the CREATE LOCK command will succeed.
3. Only UPDATE locks can be created on files in directory control directories.
4. You can lock a file in a directory control directory that is accessed R/O if no one has the directory accessed R/W or has the file locked.
5. For file control directories and files within, existing locks can affect whether 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 can cannot lock files in that directory, unless the directory is locked SHARE or UPDATE and you are trying to lock the files SHARE.
6. If you create an EXCLUSIVE lock on a file control directory and another user already has the directory accessed, their access to the directory remains in effect. Even if the user releases the directory, they will be able to reaccess it until they re-IPL or their connection to the file pool is broken.
7. You can lock a directory control directory if no one, including you, has it accessed R/W, no one else has it or a file in it locked, and you do not have it accessed R/O.

Applicable to only the byte file system (BFS)

1. Locks are created in the directory only for BFS regular files. Locks are not created for links, subdirectories, or special files.
2. CREATE LOCK may be used to create an explicit lock on a byte file system file. However, the CMS short file name format of the file name must be used. You may not enter the CREATE LOCK command using the fully qualified byte file system (BFS) path name.
3. A BFS top directory cannot be locked. If a CREATE LOCK is attempted on a BFS top directory, the command will fail.
4. A BFS file space can be locked with the FILEPOOL DISABLE FILESPACE command.
5. You must have file pool administration authority to lock a byte file system file.
6. If a BFS object is in use by an OPENVM user, it affects whether you can create a lock. A file cannot be locked if it is in use by an OPENVM user because BFS files may not be locked SHARE.
7. An EXCLUSIVE lock creates a busy file condition.

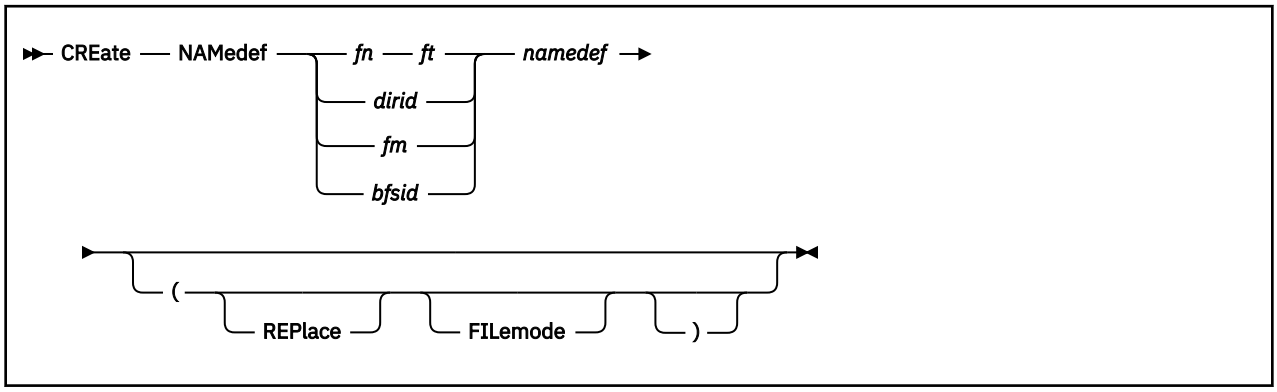
Messages and Return Codes

- DMS002E File *fn ft fm* | *dirname* not found [RC=28]
- DMS069E Filemode *mode* not accessed [RC=36]
- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS1132E Invalid number of operands [RC=24]
- DMS1163E The CREATE LOCK command failed for *fn ft fm* | *dirname* [RC=nn]
- DMS1184E Directory *dirname* not found or you are not authorized for it [RC=28]
- DMS1184E File *fn ft* or directory *dirname* not found or you are not authorized for it [RC=28]
- DMS1184E File *fn ft fm* not found or you do not have write authority to it [RC=28]
- DMS1184E Directory *dirname* not found or you do not have write authority to it [RC=28]
- DMS1187E Too many subdirectory levels in *dirname* [RC=24]
- DMS1188E Filemode *mode* is not associated with a directory [RC=74]
- DMS1212E You have opened a file pool catalog for WRITE on work unit *workunitid* for file pool *filepoolid*.
- DMS1214E You have already created a lock of type {EXCLUSIVE | SHARE | UPDATE} on file *fn ft dirname* | *fm* [RC=28]
- DMS1214E You have already created a lock of type {EXCLUSIVE | SHARE | UPDATE} on directory *dirname* [RC=28]
- DMS1215E A lock of type {EXCLUSIVE | SHARE | UPDATE} on file *fn ft dirname* | *fm* was already created by another user [RC=28]
- DMS1215E A lock of type {EXCLUSIVE | SHARE | UPDATE} on directory *dirname* was already created by another user [RC=28]
- DMS1223E There is no default file pool currently defined [RC=40]
- DMS1291E There are no unused work units available [RC=88]
- DMS2040E CREATE LOCK SHARE [or EXCLUSIVE] cannot be performed on [file *filename filetype* which is associated with] {a directory control directory|a BFS file.} [RC=24]
- DMS2133E *fn ft bfsid* is not a BFS regular file. You cannot create a lock for it. [RC=24]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411
Errors in the Shared File System	“File Pool Server Messages” on page 1414

CREATE NAMEDEF



Authorization

General User

Purpose

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
- File mode letter.
- Byte file system ID

Use QUERY NAMEDEF to display *namedefs*.

Operands

fn ft
specifies the file name and file type of the file the *namedef* represents.

dirid
specifies the directory name the *namedef* represents. For more information specifying the *dirid*, see "Naming Shared File System (SFS) Directories" on page 6.

fm
specifies the file mode letter the *namedef* represents. The file mode letter does not have to be accessed at the time the CREATE NAMEDEF command is issued. If the file mode letter is also an accessed SFS directory, the directory name will be used instead of the file mode letter when the *namedef* is created. To ensure the file mode letter is used and not the directory name, you must use the FILEMODE option.

bfsid
identifies the byte file system (BFS).

For a BFS file, *fn* and *ft* are system-generated values, which can be determined using the OPENVm LIStfile command with the NAMes SUBDirectory options specified. Together, these system-generated values uniquely identify the file within the BFS.

For more information about the OPENVM LISTFILE command, see *z/VM: OpenExtensions Commands Reference*.

namedef
specifies a 1-16 character temporary name to refer to a file name and file type or a directory name or file mode letter.

Options

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

FILemode
means the input file mode letter will be used. No resolution to an SFS directory name will be done when the *namedef* is created.

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* 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, directory, or byte file system is renamed, the *namedef* continues to refer to the original name of the *fn ft*, *dirid*, or *bfsid*, which may no longer exist.
5. If the directory name is specified as a file mode, for example,

   ```
   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 a file mode letter is specified and it is not an accessed SFS directory, the *namedef* represents the file mode letter. This will also be the case if the FILEMODE option is used. A program using this *namedef* will then be substituting a file mode letter when it calls a CSL routine that supports *namedefs*. If a file mode letter of an accessed SFS directory is specified and the FILEMODE option is not used, the *namedef* will represent the fully qualified SFS directory name accessed at the file mode letter.
7. If you are not sure which *namedefs* have already been created, use the QUERY NAMEDEF command.

Examples

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. Because the *namedefs* are resolved at run time you do not need to have them defined before compiling your program. If you wanted to specify a file mode letter as a *namedef*, you could code:

```
create namedef b fileid (filemode
```

The FILEMODE option ensures *b* is used and not an SFS directory name accessed at *b*. If the FILEMODE option is not specified and an SFS directory was accessed at mode *b*, the SFS directory name will be substituted in the *namedef*.

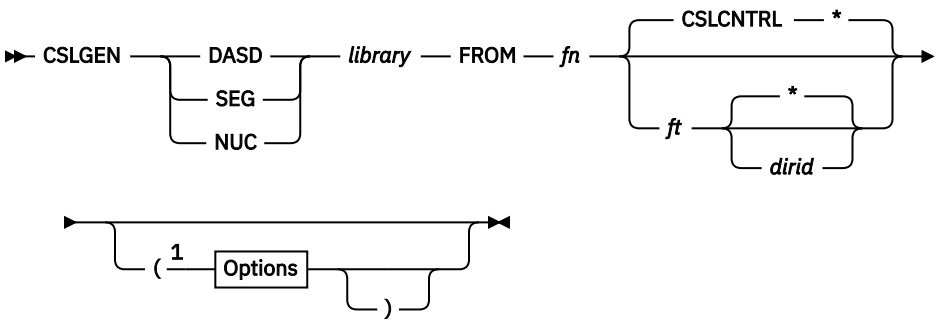
Messages and Return Codes

- DMS0659E {REPLACE|FILEMODE} option specified twice [RC=24]
- DMS069E Filemode *mode* not accessed [RC=36]
- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS1187E Too many subdirectory levels in *dirid* [RC=24]
- DMS1188E Filemode *mode* is not associated with a directory [RC=74]
- DMS1189E Filemode *mode* is associated with a top directory [RC=24]
- DMS1191E Namedef *namedef* already exists [RC=28]
- DMS1223E There is no default file pool currently defined [RC=40]
- DMS2509E FILEMODE option specified without filemode operand [RC=24]

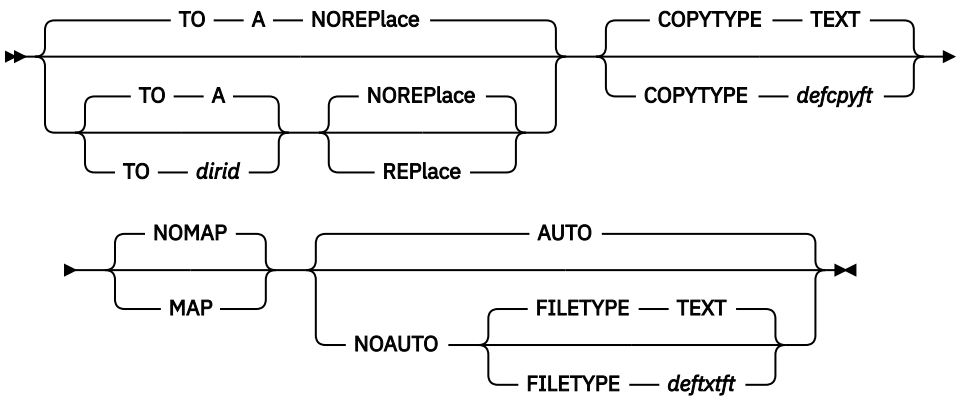
Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411
Errors in the Shared File System	“File Pool Server Messages” on page 1414

CSLGEN



Options



Notes:

[1] You can enter Options in any order between the parentheses.

Authorization

General User

Purpose

Use the CSLGEN command to build a callable services library (CSL) from control files, text files, and template files. Once you have built the library, you can store it on a common disk that users can access or you can store it in a logical saved segment. There is also a function, intended for IBM use only, that allows the library to be included within the CMS nucleus.

Operands

DASD
specifies the CSL to be built is formatted for DASD. This type of library has a file type of CSLLIB.

History information is stored directly in the CSLLIB file.

SEG
specifies the CSL to be built is formatted for use in a logical saved segment. This type of library has a file type of CSLSEG.

For more information about creating the physical and logical segment definition files and storing the library in a logical saved segment, see *z/VM: Saved Segments Planning and Administration*.

NUC
specifies the CSL to be built is formatted for inclusion within the CMS nucleus. This type of library has a file type of TEXT.

Note: This function is intended for IBM use only.

library
is the name of the CSL to be built. This library name can have a maximum of eight characters.

When CSLGEN is successful, the new CSL is placed in the directory or minidisk specified by the T0 option. The file name of the library file is *library*. The file type of the library file is CSLLIB (if the DASD operand was specified), CSLSEG (if the SEG operand was specified), or TEXT (if the NUC operand was specified). In addition if there are routines using the PATH option in the control file, a txtlib is placed in the directory or minidisk specified by the T0 option. The file name of the txtlib is *library*. The file type of the txtlib is TXTLIB. This file will contain a call routing code segment for each routine specifying a path. This file will also contain any text file additions specified by the TEXT input record.

FROM *fn*
specifies the file name of the control file for this library.

ft
specifies the file type of the control file for this library.

CSLCNTRL
specifies the control file for this library has this default file type.

dirid
identifies a directory or file mode to search to find the control file.

*****
specifies a full search, using the CMS file search order, is used to find the control file. This is the default.

Options

TO *dirid*
identifies a directory or file mode to contain the new library and the library txtlib file generated.

TO A
if T0 is not specified, the library is placed on the A-disk by default.

REPlace
indicates an existing file with the same file identifier as the new CSL should be replaced.

NOREPlace
indicates an existing file with the same file identifier as the new CSL library should not be replaced. This is the default.

FILETYPE
specifies the default file type for any text files used by CSLGEN to create the library file and the library txtlib file. The file type may be:

TEXT
is the default when FILETYPE is not specified

deftxtft
can be any valid file type.

Specifying FILETYPE *deftxtft* implies the specification of NOAUTO by default.

COPYTYPE
specifies the default file type for text files which are generated by using the COPY option on any ROUTINE or ALIAS statements within the CSLCNTRL file. For more information on the ROUTINE and ALIAS statements, see *z/VM: CMS Application Development Guide for Assembler*.

TEXT
is the default when FILETYPE is not specified

defcpyft
can be any valid file type.

NOMAP
Specifies CSLGEN will not create a map file for the library. This is the default.

MAP
Specifies a map file is to be created for the library being generated.

The file has the name:

- '*library* LIBMAP' when the DASD option is specified
- '*library* SEGMAP' when the SEG option is specified

The map file is saved on the same disk or directory as was specified for the library itself.

AUTO
Specifies CSLGEN lets the CMS loader search DASD, SFS directories, and globaled txtlibs to find text files needed to link together CSL routines. This is the default.

NOAUTO
Specifies CSLGEN directs the CMS loader using a list of text files to be taken from DASD or SFS directories, if they exist, when linking together CSL routines in the library. The CMS loader allows automatic txtlib searching to locate additional linkage parts not listed, after the listed linkage parts are included into the CSL routine. This is the default if the FILETYPE option is used. For more information on the NOAUTO option, see *z/VM: CMS Application Development Guide for Assembler*.

Usage Notes

1. If the MAP option is specified but no CSL routines are stored within the CSL library (for example when the CSLCNTRL file contains only ALIAS statements) no LIBMAP or SEGMAP file will be produced. Furthermore, if DASD was specified, the library LIBMAP file (if it exists) will be erased upon successful completion of the library build. Similarly, an existing library SEGMAP file will be erased if no routines are contained within the new library CSLSEG file.
2. If the option TO *dirid* is specified and *dirid* is not accessed, CSLGEN will search for the last available file mode and access *dirid* as that file mode.
3. A *library* TXTLIB file will only be generated if one or more ROUTINE records within the CSLCNTRL file specify a path, or if one or more TEXT or ALIAS records appear in the control file.
4. Before CSLGEN processing begins the *library* TXTLIB file is erased from the disk or directory indicated by the TO option.
5. Before CSLGEN physically builds the library (for example, performs the LOADs, INCLUDEs, and GENMODs), it saves the caller's GLOBAL TXTLIB environment and establishes its own default GLOBAL TXTLIB environment by reissuing the caller's Global TXTLIB command with *library* added at the beginning. This environment will allow any direct calls, between CSL routines within the library being built, to be resolved by the CALL routing code segments in the TXTLIB. This allows calls to CSL routines from CSL routines to be front-ended, and reduces the library size.
6. To override the default GLOBAL TXTLIB environment, create your own TXTLIB line as the first noncomment line in the CSLCNTRL file. If your CSL routines directly call CSL routines residing in a different library, you will want to add the library txtlib to the default GLOBAL TXTLIB environment.
7. To establish another GLOBAL TXTLIB environment, enter a TXTLIB line in the CSLCNTRL file. This will change the GLOBAL TXTLIB environment for all routines which follow it in the CSLCNTRL file. More than one TXTLIB line may be entered in a CSLCNTRL file.
8. To remove the GLOBAL TXTLIB environment enter a TXTLIB line containing just the keyword TXTLIB.
9. When using the TO option to target a disk or directory with a file mode other than A, the target disk or directory may be temporarily accessed as file mode A to allow creation of the CSL TXTLIB. This occurs if the library will contain direct call routines or aliases, or if one or more text records were specified in the control file.

When the original disk or directory is restored in the A file mode, it will be reaccessed with all files that are generally available. If the disk or directory was originally accessed with only a subset of files available, for example

```
ACCESS 191 A/A * EXEC
```

it will be reaccessed with the equivalent of the command

```
ACCESS 191 A/A
```

and may result in additional files being exposed to the CMS search order.

10. If routines are defined within the CSLCNTRL file using the CSECT option and the SEGMENT option on the CSLGEN command is specified, warning messages from the INCLUDE command (messages resulting in a return code of 4) are ignored. For more information on the CSECT option, see *z/VM: CMS Application Development Guide for Assembler*. The return code of 4 is returned by CSLGEN so messages from INCLUDE were issued. These messages should be saved and reviewed to ensure the library has been properly built.
11. The COPYTYPE option can be overridden for individual routine and alias entries by using the COPYTYPE option on the ROUTINE and ALIAS statements in the CSLCNTRL file.
12. The LIBMAP and SEGMAP load maps produced when the MAP option is specified are adapted from the load map created by the CMS LOAD command using the FULLMAP option. In addition to the original load map information, the CSL library load map displays CSL routine names with their corresponding CSECT labels. For examples of CSL library load maps, see *z/VM: CMS Application Development Guide for Assembler*.
13. During CSLGEN processing any direct call stubs created are temporarily held within the $$CSL$$ TXTLIB file. The existing library TXTLIB file is not replaced until successful completion of the library build. Any appearance of the library name within a GLOBAL TXTLIB command issued by CSLGEN will be translated to $$CSL$$ so the latest stub changes will be used.
14. History information is kept as follows:
    - When DASD is specified
      - All noncommented history is stored directly in the *library* CSLLIB file.
      - All noncommented history is stored in the *library* LIBMAP file when the MAP option is specified.
      - All commented history is stored in the individual routine LOADMAP files generated by using the MAP option on the ROUTINE line.
    - When SEG is specified
      - All noncommented history is stored in the *library* SEGMAP file when the MAP option is specified.
      - All noncommented history is stored in the LOAD MAP file when MAP is not specified.
    - When NUC is specified
      - All history is stored in the nucleus map.
15. Specifying the MAP option will cause the CMS LOAD and INCLUDE commands to store any load map files they produce on the A-disk. Therefore the A-disk must be read-write.
16. When the NOAUTO option is specified, each ROUTINE statement in the CSLCNTRL file must be followed by the list of additional text files that can come from DASD or SFS directories and are needed to complete the CSL routine. The INCLUDE statement lists the files to be included. For more information on the INCLUDE statement and its use, see *z/VM: CMS Application Development Guide for Assembler*.
17. When AUTO is specified, all INCLUDE statements within the CSLCNTRL file are ignored.
18. When AUTO is specified, all alternate file types specified on ROUTINE, TEXT, and INCLUDE statements within the CSLCNTRL file are ignored. A file type is not passed to the CMS loader.
19. When NOAUTO is specified and a text file is named on more than one ROUTINE or INCLUDE statement, the file type must be the same for all occurrences.

20. When the NUC operand is specified, CSLGEN does not check for the existence of the text files listed in the CSLCNTRL file. All text files are assumed to already reside within the CMS nucleus. However, the text file names and alternate CSECT labels within the CSLCNTRL file are still needed to set the proper linkage within the library.
21. The NOAUTO, AUTO, FILETYPE, and MAP options are ignored when the NUC operand is specified.
22. When the NUC operand is specified, the copies of the library TXTLIB stubs differ from the TXTLIB members. The copies are modified for inclusion within the CMS nucleus. These copies should not be used by application programs. The TXTLIB members are intended for use by application programs and are the same as those produced when the DASD or SEG operand is specified.
23. When the NUC operand is specified, all direct call paths used by the library routines must be unique, such that each path used is defined on, at most, one ROUTINE record. If more than one library will reside in the nucleus, then all paths defined in all the libraries should be unique.

Examples

In the following example, a CSL named NEWLIB CSLLIB is built by a CSLGEN command. NEWLIB CSLLIB contains nine routines, and it is placed in user ID JOHNDOE's JD.TEST directory on the system with file pool ID FPID1. Here is what the CSLGEN command looks like:

```
cslgen dasd newlib from newcon (to fpid1:jd.test
```

Four control files are used to build the library. NEWCON CSLCNTRL, located on JOHNDOE’s A-disk, is specified on the CSLGEN command line. It looks like this:

```
ROUTINE rtn1
ROUTINE rtn2
ROUTINE rtn3
ROUTINE rtn4
CSLCNTRL math cntrl fpid2:janedoe.jd.test
*
* NEWCON CSLCNTRL specifies 4 routines to be loaded into the library.
* For each routine, template file name = text file name = routine name;
*  template file type is TEMPLATE.
* This also specifies one additional control file.
```

The additional control file, MATH CNTRL, contains the following:

```
CSLCNTRL math part1
CSLCNTRL math part2
CSLCNTRL math part3
*
* This is the control file for all MATH routines;
*  it specifies three other, separate control files.
```

MATH PART1 contains these lines:

```
ROUTINE xxx rtn5 rtn5tmpl
ROUTINE yyy rtn6 rtn6tmpl
ROUTINE zzz rtn7 rtn7tmpl
*
* This is the first MATH control file.
* It specifies 3 routines to be loaded into the library; each
*  routine will be loaded under a name different than its TEXT name.
* The file type of the template files is TEMPLATE.
```

and MATH PART2 contains these lines:

```
ROUTINE aaa rtn8 rtn8tmpl templ
ROUTINE bbb rtn9 rtn8tmpl templ
```

and MATH PART3 contains these lines:

```
ROUTINE abc rtn10 rtn10tmp (PATH 1.27
ROUTINE axy rtn11 rtn10tmp (PATH 1.28
ALIAS xyz PATH 1.30
```

Table 8 on page 116 shows the routine names, TEXT files, and template file used to build the library NEWLIB CSLLIB:

Table 8. Building a NEWLIB CSLLIB

Routine Name	TEXT File Name	Template File Name	Template File type
RTN1	RTN1	RTN1	TEMPLATE
RTN2	RTN2	RTN2	TEMPLATE
RTN3	RTN3	RTN3	TEMPLATE
RTN4	RTN4	RTN4	TEMPLATE
XXX	RTN5	RTN5TMPL	TEMPLATE
YYY	RTN6	RTN6TMPL	TEMPLATE
ZZZ	RTN7	RTN7TMPL	TEMPLATE
AAA	RTN8	RTN8TMPL	TEMPL
BBB	RTN9	RTN8TMPL	TEMPL
ABC	RTN10	RTN10TMP	TEMPLATE
AXY	RTN11	RTN10TMP	TEMPLATE

The new load map format created by CSLGEN follows:

```
Routine  Name  Ty Origin   Length                        Fn    Ft     Fm   Member Timestamp
RTNNAME3 PROG1 SD 00020000 00000318 RMODE 24  AMODE 24  PROG1 TEXT   A1          1/11/91  14:37:41
         PROG2 SD 00020318 00000830 RMODE ANY AMODE 31  PROG2 TEXT   A1          1/11/91  14:37:50
         PROG3 SD 00020B48 00000210 RMODE 24  AMODE ANY LIB1  TXTLIB A1   PROG3  12/26/90 15:25:30
```

Figure 2. CSLGEN's load map format

Routine is the CSL routine name associated with the CSECT name listed under the NAME column. For information on the other columns, see “LOAD” on page 471.

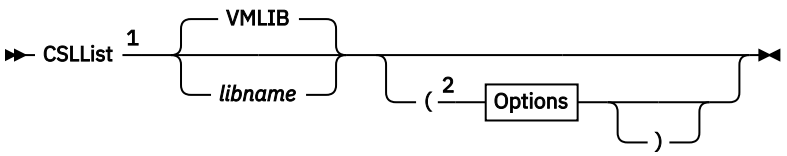
Messages and Return Codes

- DMS002E File *fn ft fm* not found [RC=28]
- DMS024E File *fn ft fm* already exists; specify REPLACE option [RC=28]
- DMS037E Filemode *fm* is accessed as read/only; *fm* must be R/W for CSLGEN [RC=36]
- DMS037I *fn ft fm* could not be erased; filemode *fm* is read-only [RC=36]
- DMS056E File *fn ft fm* contains invalid record formats [RC=32]
- DMS056E File *fn ft fm* contains invalid [*ROUTINE* | *ALIAS* | *TEXT* | *TXTLIB* | *CSLCNTRL* | *INCLUDE*] record formats [RC=32]
- DMS065E *option* option specified twice [RC=24]
- DMS065E Unexpected *unexpected string* allowed by the parser. The level of CSLGEN may not correspond to the level of CMS [RC=24]
- DMS065E Unexpected *unexpected string*. CSLGEN error. [RC=24]
- DMS066E *opt1* and *opt2* are conflicting options [RC=24]
- DMS066E Conflicting options *opt1* and *opt2* specified for routine *rtnname*. [RC=24]
- DMS069E Filemode *fm* not accessed [RC=36]
- DMS1090E Invalid CSL path *path* specified [RC=24]
- DMS1091E Error reading from file *fn ft fm*; EXECIO rc=*retcode* [RC=26]

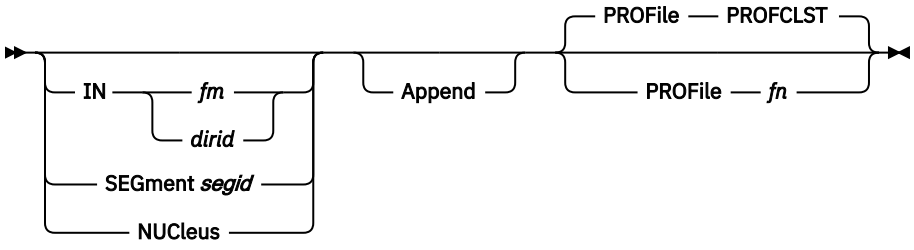
- DMS1092E Error writing to an intermediate CSL file; EXECIO rc=*retcode* [RC=26]
- DMS1094E CSL control file must not have filetype "TEXT" [RC=24 or 32]
- DMS1094E TXTLIB extension record does not start with the string "TXTLIB" [RC=24]
- DMS1094E More than 63 txtlibs have been specified on a TXTLIB record and its continuation TXTLIB records [RC=24]
- DMS1096E Duplicate [*ROUTINE*|*ALIAS*|*TEXT*] name specified in the control files [RC=28]
- DMS1096E *name* used in both a [*ROUTINE*|*ALIAS*|*TEXT*] record and a [*ROUTINE*|*ALIAS*|*TEXT*] record [RC=28]
- DMS1096E 'INCLUDE *fn*' statement is not preceded by a ROUTINE or INCLUDE line [RC=28]
- DMS1096E Text file *fn* appears on more than one statement and the file type is not the same for all occurrences [RC=28]
- DMS1096E *rtnname* used as both a TEXT file name and as a direct call ROUTINE name [RC=28]
- DMS1096E Path *path* used twice in a nucleus resident library [RC=28]
- DMS1100E No filemode is available to access *dirid* [RC=28]
- DMS1108I CSLGEN completed. Library *fn ft fm* built with TXTLIB. [RC=0]
- DMS1108I CSLGEN completed. Library *fn ft fm* built with TXTLIB. Size = *bytes* [RC=0]
- DMS1108I CSLGEN completed. Library *fn ft fm* built. [RC=0]
- DMS1108I CSLGEN completed. Library *fn ft fm* built. Size= *bytes* [RC=0]
- DMS1109I CSLGEN terminated. No library built.
- DMS1110E CSLGEN encountered error executing *command*, rc=*retcode* [RC=40 from CSLGEN]
- DMS1111I GLOBAL TXTLIB environment *environment* could not be restored [RC=4]
- DMS1111I DMSHSH routine could not be NUCXDROPped [RC=4]
- DMS1111E No routines or aliases are specified in the CSL control files [RC=28]
- DMS1111E Invalid file *fn ft fm* [RC=28]
- DMS1112E Duplicate control file *fn ft* specified in the CSL control files [RC=28]
- DMS1235E Template *template* in *fn ft fm* is invalid [RC=28]
- DMS1235E Length value for *datatype* on line *linenum* in file *fn ft fm* is invalid [RC=28]
- DMS1235E Direction value for *datatype* on line *linenum* in file *fn ft fm* is invalid [RC=28]
- DMS1235E *datatype* record, line *linenum* in file *fn ft fm* has no matching TABLE record [RC=28]
- DMS1235E *datatype* record, line *linenum* in file *fn ft fm* cannot be used alone [RC=28]
- DMS1235E *datatype* record, line *linenum* in file *fn ft fm* cannot be followed by a *datatype2* record [RC=28]
- DMS1235E TABLE record, line *linenum* in file *fn ft fm* has no defined columns [RC=28]
- DMS1235E *datatype* record, line *linenum* in file *fn ft fm* has no TABLE definition [RC=28]
- DMS1235E *datatype* record, line *linenum* in file *fn ft fm* cannot have an associated OUTPUT length [RC=28]
- DMS1235E *datatype* record, line *linenum* in file *fn ft fm* has too many associated *direction* LEN records [RC=28]
- DMS1235E *datatype* record, line *linenum* in file *fn ft fm* has no associated INPUT LEN record [RC=28]
- DMS1235E *datatype* record, line *linenum* in file *fn ft fm* has a direction which conflicts with the table direction [RC=28]
- DMS1235E *datatype* record, line *linenum* in file *fn ft fm* is an optional parameter associated with the required *datatype2* parameter [RC=28]
- DMS1235E *datatype* record, line *linenum* in file *fn ft fm* is both a TABLE column and an indirectly addressed parameter [RC=28]

- DMS1235E Direction value for *datatype* on line *linenum* in file *fn ft fm* conflicts with direction for *datatype2* on line *linenum* [RC=28]
- DMS1235I Invalid length value for [PTR|RTNV|RTNC|RTNR] record, line *linenum* in file *fn ft fm*, is reset to 4 [RC=4]
- DMS1235I Invalid direction value for [RTNV|RTNC|RTNR] record, line *linenum* in file *fn ft fm*, is reset to OUTPUT [RC=4]
- DMS1235I Duplicate [RTNV|RTNC|RTNR] records, in file *fn ft fm*. [RC=28]
- DMS1235I Duplicate [RTNV|RTNC|RTNR] records, in member *membername* in file *fn ft fm*. [RC=28]
- DMS1236E Invalid specification in *fn ft fm* of the number of templates [defining required parameters] [RC=28]
- DMS1236E Invalid specification in member *membername* in file *fn ft fm* of the number of templates [defining required parameters] [RC=28]
- DMS1236E Missing DIRECT or OPENVM keyword in template file *fn ft fm* for routine *name*. The [*PATH*|*MP*] option is specified but the DIRECT or OPENVM keyword is missing in the template [RC=28]
- DMS1236E Missing DIRECT or OPENVM keyword in template member *membername* in file *fn ft fm* for routine *name*. The [*PATH*|*MP*] option is specified but the DIRECT or OPENVM keyword is missing in the template file. [RC=28]
- DMS1236E Invalid specification of return code position [RC=28]
- DMS1237E Line *line number* in file *fn ft* is an invalid definition of return code data [RC=28]
- DMS1237E The first template in *fn ft fm* is an invalid definition of return code data [RC=28]
- DMS1237E The first template in template member *membername* in file *fn ft fm* is an invalid definition of return code data [RC=28]
- DMS2055I [*COPY*|*COPYTYPE*|*PROTECT*|*MAP*] option ignored for routine *rtnname* [RC=4]
- DMS2542I Temporarily accessing target disk/directory as file mode *fm* [RC=0]
- DMS2543I Restoring original CMS search order [RC=0]

CSLLIST



Options



Notes:

[1] A special command that can be used in the CSLLIST environment is EXECUTE.
[2] You can enter Options in any order between the parentheses.

Authorization

General User

Purpose

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

Operands

libname
specifies the callable services library to be listed. The default is VMLIB.

If the location of the library is not specified by using the IN, SEGMENT, or NUCLEUS option, saved segments are searched first, the CMS nucleus is searched second, 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. For more information specifying the *dirid*, see “Naming Shared File System (SFS) Directories” on page 6.

SEGment *segid*
identifies the saved segment containing the library.

NUCleus
specifies the library is included in the CMS nucleus.

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

Usage Notes

1. For more information on how to customize this command, see Appendix A, “Customizing Profiles for CMS Productivity Aids,” on page 1419.
2. 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. For more information, see “EXECUTE” on page 1393.
3. When you invoke CSLLIST you are under control of 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, or NUCLEUS for CMS nucleus)
   - The library name
   - The logical segment name or file type
   - The file mode or a blank if the library resides in a segment
   - The path if the routine is a direct call CSL routine
   - Other attributes defined for the routine
4. Begin CMS commands with "CMS" to prevent XEDIT from decoding the command. This prevents CMS commands from being mistaken as XEDIT subcommands.
5. The DEFAULTS command can be used to set up options and override command defaults for CSLLIST. The options 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.
6. You can issue commands directly from the line that displays a routine by simply typing the command in the command (Cmd) column of that line.

   If a command is longer than the command column, just continue typing over the information in the line; you may type over the entire line displayed up to column 74.

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

   / (or /n)
   means the CSL routine name displayed on the line.

   /g
   means the subgroup name displayed on the line.

/l
means the library name 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.

8. 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 the **?** was entered.

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

9. 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 default key settings set by PROFCLST XEDIT are shown in Table 9 on page 121.

Table 9. Default Key Settings of PROFCLST XEDIT

Key	Setting	Action
Enter	Execute	Executes command(s) typed on routine line(s) or on the command line.
PF 1	Help	Displays CSLLIST command description.
PF 2	Refresh	Updates the list to indicate new routines, erased routines, and so forth, using the same parameters as those specified when CSLLIST was invoked.
PF 3	Quit	Exits from CSLLIST.
PF 4	Template	Displays template information for that routine.
PF 5	Sort (routine)	Sorts alphabetically by routine name.
PF 6	CSLMAP	Issues a CSLMAP /a (ALL command for the routine selected by the cursor to display any loaded versions of that routine. CSL routines may be dropped from CSLMAP.
PF 7	Backward	Scrolls back one screen.
PF 8	Forward	Scrolls 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 *fm* is issued.
PF 10	Attribute	Displays assigned attributes.
PF 11	Sort (libname)	Sorts the list alphabetically by library name and then by routine name.
PF 12	Cursor	Moves it to the command line if the cursor is in the list area. Moves it back to its previous location in the list (or to the current line) if the cursor is on the command 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 previous PF keys, the PROFCLST XEDIT macro sets synonyms 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.

10. The following commands are related to CSLLIST:
    - CSLMAP – displays information about currently loaded CSL routines
    - RTNLOAD – loads a CSL routine
    - RTNMAP – lists the CSL routines currently loaded
    - RTNDROP – drops a loaded CSL routine
11. 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.
12. For more information on how to customize your user PROFILE, see Appendix A, "Customizing Profiles for CMS Productivity Aids," on page 1419.

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 six routines that each generate a description for a class. If the user enters the command

```
csllist courses (in a
```

the following might be displayed:

```
 JOHNDOE  CSLLIST  A0  V 225  Trunc=225 Size=5  Line=1 Col=1 Alt=0
Cmd   Routine  Resident  Libname  Segid/Type  Mode    Path    Subgroup
      GEOGRAPH DASD      COURSES  CSLLIB      A1
      HISTORY  DASD      COURSES  CSLLIB      A1    0513.250
      PHYSICS  DASD      COURSES  CSLLIB      A1    0001.47   SCIENCE
      PSYCH    DASD      COURSES  CSLLIB      A1
      CALCULUS DASD      COURSES  CSLLIB      A1    0001.97   MATH
      MATH     DASD      COURSES  CSLLIB      A1    0513.254  MATH









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

====>
```

How to Read the Path Display Screen

The following list shows you what information some of the columns on the screen provide:

Path
shown for routines HISTORY, PHYSICS, CALCULUS, and MATH indicate these routines can be called using either the

```
CALL rtnname (...
```

or

```
CALL DMSCSL(rtnname,...
```

formats. GEOGRAPH and PSYCH, not showing paths, can only be called using the 'CALL DMSCSL(rtnname,...' interface.

Subgroup
subgroupings of routines within the library are shown under the Subgroup heading. Two subgroups exist in this library. The routines CALCULUS and MATH belong to the subgroup MATH in this library. This subgroup name can be used at RTNLOAD and RTNDROP to load or drop only the routines within a specific subgroup within a specific library. For more information, see "RTNDROP" on page 840 and see "RTNLOAD" on page 843.

PF Keys
When PF 11 is pressed, a sort will be done on two columns, primarily on the Libname column and secondarily on the Subgroup column.

When PF 6 is pressed a 'CSLMAP /n (ALL' command is issued. Dropping CSL routines can be performed from within CSLMAP.

Pressing PF 10 (Attribute) takes you to the next display screen where the assigned attributes are displayed.

All other PF key functions are unchanged.

2. From the previous CSLLIST environment, when PF 10 is pressed the following will be displayed:

```
 JOHNDOE  CSLLIST  A0  V 225  Trunc=225 Size=5  Line=1 Col=1 Alt=0
Cmd   Routine  Libname  Segid/Type  Alias  Interface  Protect
      GEOGRAPH COURSES  CSLLIB        -        1         -
      HISTORY  COURSES  CSLLIB        -        -         -
      PHYSICS  COURSES  CSLLIB        -        -         -
      PSYCH    COURSES  CSLLIB        -        1         -
      CALCULUS COURSES  CSLLIB        -        1         -
      MATH     COURSES  CSLLIB        X        -         -









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

====>
```

How to Read the Attribute Display Screen

The following list shows you what information some of the columns on the screen provide:

Heading
Description

Alias

X
indicates *rtnname* was defined as an alias using an ALIAS input record in the CSLCNTRL file used to create *libname*.

-
indicates *rtnname* was defined as a routine.

Interface
shows the type of interface expected by the routine:

-
The routine requires the indirect CSL interface. It can only be called by the DMSCSL, CSLFPI, and the REXX CSL interfaces.

1 or 1+
The routine is directly callable. It may be called using the DMSCSL, CSLFPI, and REXX CSL interfaces. It may also be called directly if it has an entry under the Path column. The differences between 1 and 1+ are as follows:

1
The routine is not multiprocessor capable. The caller is required to provide a register save area when making DMSCSL and direct calls.

1+
The routine is multiprocessor capable (MP). The caller is required to provide a register save area when making DMSCSL calls. No save area is needed when making direct calls to CMS MP routines.

2, 2+, 2*, or 2+*
The routine uses a nonstandard parameter list. This routine can be called only directly or by the ADDRESS OPENVM REXX interface.

+
The routine is multiprocessor (MP) capable, and therefore a save area is not required when making calls to the routine.

***
The routine's parameter list contains no parameter in which a value can be returned to indicate the direct call interface failed to invoke the routine. A direct call interface error causes CSL to initiate an ABEND. An interface error while calling with the ADDRESS OPENVM interface does not result in an ABEND. In that case, the interface error is returned in the REXX variable RC.

Protect

X
indicates the routine is protected; after the initial RTNLOAD, subsequent RTNLOAD or RTNDROP requests will not be allowed.

D
indicates the routine is protected from being dropped; after the initial RTNLOAD, subsequent RTNDROP requests will not be allowed. However, RTNLOAD can be used to replace the active version of the routine.

-
indicates the routine is not protected.

If you mark a routine protected, it will be ignored if the routine is not destined for a segment.

Pressing PF 10 (Path) will return you to the previous display screen which displays the path and Subgroup information.

3. 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 225  Trunc=225 Size=5  Line=1 Col=1 Alt=0
Cmd   Routine  Resident  Libname  Segid/Type  Mode    Path    Subgroup
      SINE     SEGMENT   TRIG      JDSEG1
      COSINE   SEGMENT   TRIG      JDSEG1
      TANGENT  SEGMENT   TRIG      JDSEG1









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

====>
```

4. From within the CSLLIST environment shown previously in example 3, JOHNDOE can issue the command

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

   to append information about his COURSES library.

```
 JOHNDOE  CSLLIST  A0  V 225  Trunc=225 Size=5  Line=1 Col=1 Alt=0
Cmd   Routine  Resident  Libname  Segid/Type  Mode    Path    Subgroup
      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= Cslmap
7= Backward 8= Forward 9= Rtnload 10= Attribute 11= Sort(libname) 12= Cursor

====>
```

5. When you place the cursor in the command area of a listed routine and press PF4, you will get a screen that gives you *template information* for that routine.

   This screen helps you interpret the routine's template file, which contains information about what kind of parameters the routine expects. For more information on the CSL routine templates, see *z/VM: CMS Application Development Guide for Assembler*.

   For example, if you place the cursor beside the routine SINE in the previous example screen and press PF4, you would see:

```
 SINE     $TEMPLAT  A0  F 80   Trunc=80  Size=11 Line=1 Col=1 Alt=0
Record   Parameter   Direction   Datatype    Length   Units   Note
   9         1+         OUT      SBIN             4   Bytes
  10         2+         IN       SBIN             2   Bytes
  12         3+         IN       SBIN             2   Bytes
  14         4+         OUT      SBIN             4   Bytes
  21                    INOUT    TABLE            *   Rows
  22         5+         IN       -LEN             7   Bits
  25         6+         OUT      -LEN             7   Bits
  26         7+         IN       -C.SBIN          2   Bits
  29         8+         OUT      -C.SBIN          4   Bits
-------------------------------------------------------------------------
  33         9          *        CHAR             8   Bytes

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

====> _
```

Note the new set of PF keys at the bottom of the screen. This screen is controlled by XEDIT, so when you are done viewing the template information, you can enter QUIT on the command line or press PF3 to return to the CSLLIST environment.

How to Read the Template Display Screen

This screen translates the template file into information about the parameter list required when calling the routine. The following list shows you what information each column on the screen provides:

Heading
Description

Record
is the record number in the original template file. (If this column is blank, it means this parameter is implied by a previous one.)

Parameter
is the parameter's order in the parameter list. If this column is blank, it means this parameter is implied by a previous one. A plus sign (+) after the number means the parameter is required. The dashed horizontal line you may see across your screen (and as shown in the example screen) divides the required parameters (above the line) from the optional parameters (below the line).

Direction
is the direction the data in the parameter moves between the calling routine and the called routine. It may be IN, OUT, INOUT, or * (no direction; passed by a following length parameter IN to the called routine.)

Datatype
is the form in which the data in the parameter must be supplied to the routine. A '-' prefixes datatypes associated with previous non-hyphenated datatypes. For example in the SINE $TEMPLAT display above, parameters 5 - 8 are associated with the TABLE datatype preceding them. For more information on the various data types, see *z/VM: CMS Application Development Guide for Assembler*.

Length
is the expected length of the data in the parameter. The length may be:

- A non-zero decimal number.
- A 0 indicating the actual length is passed by the previous parameter.
- An ‘*’ specifying a variable length. The actual length is passed in the next parameter with a data type of LEN.
- A ‘-’ specifying a null terminated string, with an end marked by a X'00' byte following the last byte of the string.

Units
are the length units (bits, bytes, or rows).

Note
may contain a ‘1’, ‘2’, or ‘3’, referring to the following notes:

1. This parameter is implied by a previous parameter.
2. This is a length parameter implied by a previous fixed-length character parameter, and must be specified. If you are calling DMSCSL or REXX, specify it as a parameter at this position in the parameter list. If you are using CSLFPI, specify it along with the name and value of the preceding parameter using CSLFPI TYPE=SET,PARMS=*(name, value, length)*.
3. This is an indirectly addressed variable whose address must be supplied as the parameter when calling the routine with DMSCSL or CSLFPI. If calling the routine using REXX, the name of the variable is supplied instead; REXX will supply the correct pointer to the routine automatically.

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 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
```

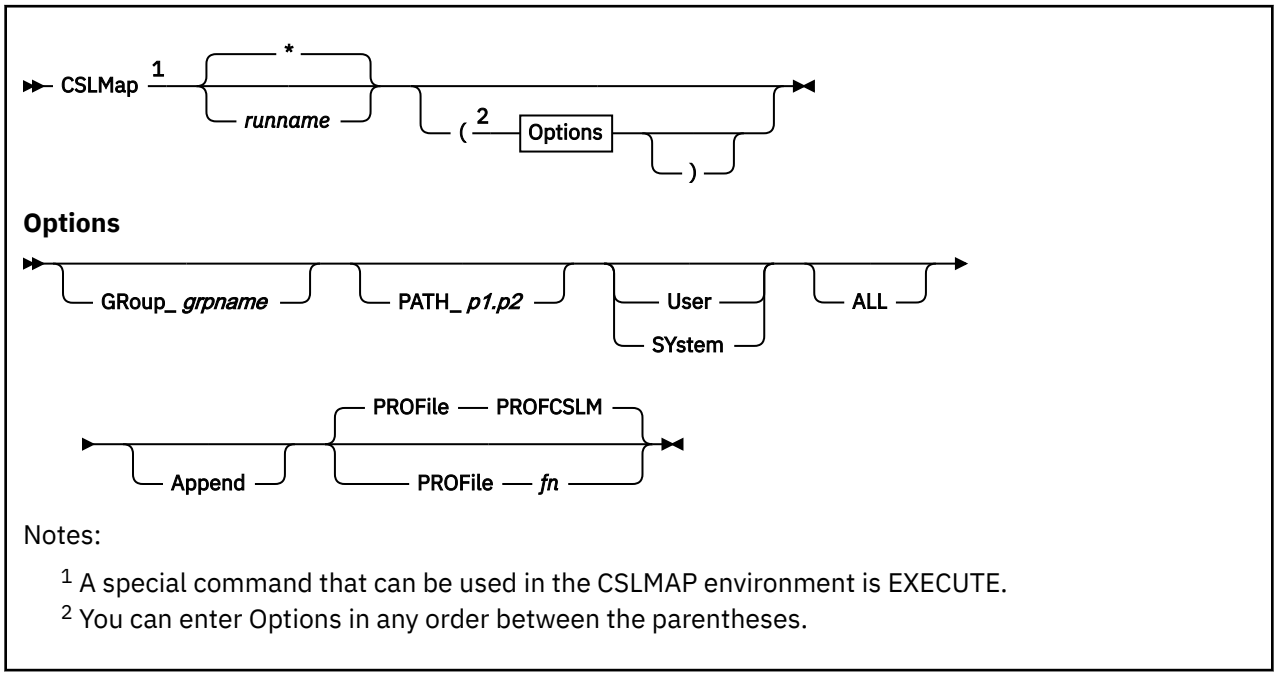
Messages and Return Codes

- DMS065E *option* option specified twice [RC=24]
- DMS066E *opt1* and *opt2* are conflicting options [RC=24]
- DMS349E Invalid library *libname* [RC=32]
- DMS639E Error in *rtnname* routine; return code was
- DMS651E APPEND must be issued from CSLLIST [RC=40]
- DMS1097E Routine *rtnname* not found [RC=28]
- DMS1100E No filemode is available to access *dirname* [RC=28]
- DMS1110E CSLGEN encountered error executing *command*, rc=*retcode* [RC=40 from CSLGEN]
- DMS1136E Unable to gain access to library *libname* [RC=28]
- DMS2506E *rtnname* is an ALIAS, no template is found [RC=28]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411
Errors in the Shared File System	“File Pool Server Messages” on page 1414

CSLMAP



Notes:

[1] A special command that can be used in the CSLMAP environment is EXECUTE.
[2] You can enter Options in any order between the parentheses.

Authorization

General User

Purpose

Use the CSLMAP command to display information about currently loaded callable services library (CSL) routines.

Operands

runname
specifies 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.

*****
specifies information is to be shown about all of the routines that satisfy the specified options.

Options

GRoup *grpname*
specifies only routines that were loaded into the specified *grpname* (on the RTNLOAD command) are to be displayed.

PATH *path*
specifies only routines that were loaded and are using the given path are to be displayed. *Path* can be *p1.p2* or '*', where '*' specifies only routines loaded with a path (the path value makes no difference in this case) are to be displayed.

p1 can be entered as:

- An integer between 1 and 1024

- '*' signifying all values between 1 and 1024 are to be used in the search.

p2 can be entered as:

- An integer between 1 and 250
- '*' signifying all values between 1 and 250 are to be used in the search.

User
specifies information is displayed only for routines loaded with the USER option (on the RTNLOAD command). For more information, see Usage Note "3" on page 129.

SYstem
specifies information is displayed only for routines loaded with the SYSTEM option (on the RTNLOAD command). For more information, see Usage Note "3" on page 129.

ALL
specifies information is displayed about all routine versions that meet the specified criteria. If ALL is not specified, information is displayed only for the most recently loaded version of the routine(s) specified by the *runname* operand.

The most recently loaded version of a routine is marked by a '>'.

Append
specifies the routine mapping to be generated is to be added to those currently being displayed. This option is only valid from within a CSLMAP full-screen session.

PROFile *fn*
specifies an XEDIT profile named "*fn* XEDIT" should be used when entering the CSLMAP environment. If this option is not used, a profile named PROFCSLM XEDIT is used.

Usage Notes

1. For more information on how to customize this command see Appendix A, "Customizing Profiles for CMS Productivity Aids," on page 1419.
2. You can use the special command EXECUTE from the CSLMAP screen. The EXECUTE command allows you to issue commands that use the routines displayed by CSLMAP. For more information, see "EXECUTE" on page 1393.
3. If neither the USER nor SYSTEM option is specified, CSLMAP displays information according to the *runname* operand and, if specified, the GROUP, ALL and PATH options.
4. When you invoke CSLMAP you are under control of XEDIT. At this time, you are editing the following CMS file:

   fn =
   user ID you are logged on with

   ft =
   CSLMAP

   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
   - The alias or *runname* of the routine
   - The original CSL routine name from the CSL library
   - The library name the routine was loaded from
   - The group the routine belongs to
   - The path, if the routine is using the CSL direct call interface
   - Other attributes defined for the routine
5. Entering CMS commands from CSLMAP

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

6. Tailoring the CSLMAP Command Options

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

7. Issuing Commands from the CSLMAP Screen

   You can issue commands directly from the line that displays a routine by simply typing the command in the command (Cmd) area of that line.

   If a command is longer than the command area, just continue typing over the information in the line; you may type over the entire line displayed.

   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.

8. 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:

   /a
   means the alias name or run name that is displayed on the line.

   / (or /n)
   means the routine name that is displayed on the line.

   /g
   means the group name that is displayed on the line.

   /p
   means the path that is specified on the line.

   /o
   means the execution of the line typed prior to the '/o' as written.

9. Special Symbols Used Alone

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

   =
   means execute the previous command for this routine. The command is executed starting from the top of the screen.

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

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

10. Default Key Settings

    Entering the CSLMAP command executes the PROFCSLM XEDIT macro, unless you specify a different macro as an option in the CSLMAP command. For more information on using another profile, see “DEFAULTS” on page 153. The PF keys are set by this macro. The default key settings set by PROFCSLM XEDIT are shown in Table 10 on page 130.

Table 10. Default Key Settings of CSLM

Key	Setting	Action
ENTER	Execute	Executes command(s) typed on routine line(s) or on the command line.

Table 10. Default Key Settings of CSLM (continued)

Key	Setting	Action
PF1	Help	Displays CSLMAP command description. For more information on the template display screen, see "CSLLIST" on page 119.
PF2	Refresh	Updates the list to indicate new routines, erased routines, and so forth, using the same parameters as those specified when CSLMAP was invoked.
PF3	Quit	Exits from CSLMAP.
PF4	Template	Displays template information for the routine specified by the cursor. For more information on the template display screen, see "CSLLIST" on page 119.
PF5	Sort (alias)	Sorts primarily on the Alias column and secondarily in the inverse order in which the routine versions were loaded.
PF6	Drop	Drops the routine version specified by the cursor. This drop routine is different from RTNDROP in that it resolves individual versions which RTNDROP cannot. For example the display shown in Example 1 shows that the same routine was loaded from the same library two different times. A RTNDROP on FORMULA could not drop the very first version loaded, but only the current version (marked by the '>'). The Drop function can make this distinction. This drop cannot remove a protected routine version.
PF7	Backward	Scrolls back one screen.
PF8	Forward	Scrolls forward one screen.
PF10	Attributes	Changes the display to the screen that displays the routine attributes.
PF11	Sort (group)	Sorts the list alphabetically by the group name column.
PF12	Cursor	Moves the cursor to the command line if the cursor is in the list area. Moves it back to its previous location in the list (or to the current line) if the cursor is on the command 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 PROFCSLM XEDIT macro sets synonyms you can use to sort your CSLMAP display. The synonyms are:

SALIAS
Sorts the list primarily by alias and secondarily in reverse order of routine version loading.

SGROUP
Sorts the list alphabetically by group name.

11. These commands are related to CSLMAP:

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

- RTNDROP – drops a loaded CSL routine
- CSLLIST – displays contents of CSL

12. Currently active routine versions are marked to the left of the alias column with these:

    >
    Marks the most recently loaded routine version. This version will be invoked when a CALL DMSCSL is executed. This is also the routine version invoked by a CSLFPI macro call.

    +
    Marks the routine version which will be invoked when a direct call like "CALL FORMULA(..." is performed. There exists only one + for each path. It marks the currently active routine version accessed by that path.

Examples

1. Suppose someone with the user ID JOHNDOE wants to view all routine versions RTNLOADed with run name 'FORMULA'. If the user enters the command

   ```
   CSLMAP FORMULA ( ALL
   ```

   the following might be displayed:

```
 JOHNDOE  CSLMAP  A0  V 130  Trunc=130 Size=4  Line=1 Col=1 Alt=0
Cmd      Alias     Name      Library   Address       Bytes  Group     Path
      >+ FORMULA   FORMULA   TEST      005CBC68        300  GROUP123  0105.103
         FORMULA   FORMULA   OLDLIB    005CF890        450
       + FORMULA   FORMULA   NEWLIB    005EC280        550  FUNCTB    0020.003
         FORMULA   FORMULA   MIGLIB    005DD000        330
         FORMULA   FORMULA   TEST      005CBC68        300  GROUP123  0105.103









1= Help     2= Refresh 3= Quit    4= Template     5= Sort(Alias)    6= Drop
7= Backward 8= Forward 9=        10= Attributes  11= Sort(Group)   12= Cursor

====>
```

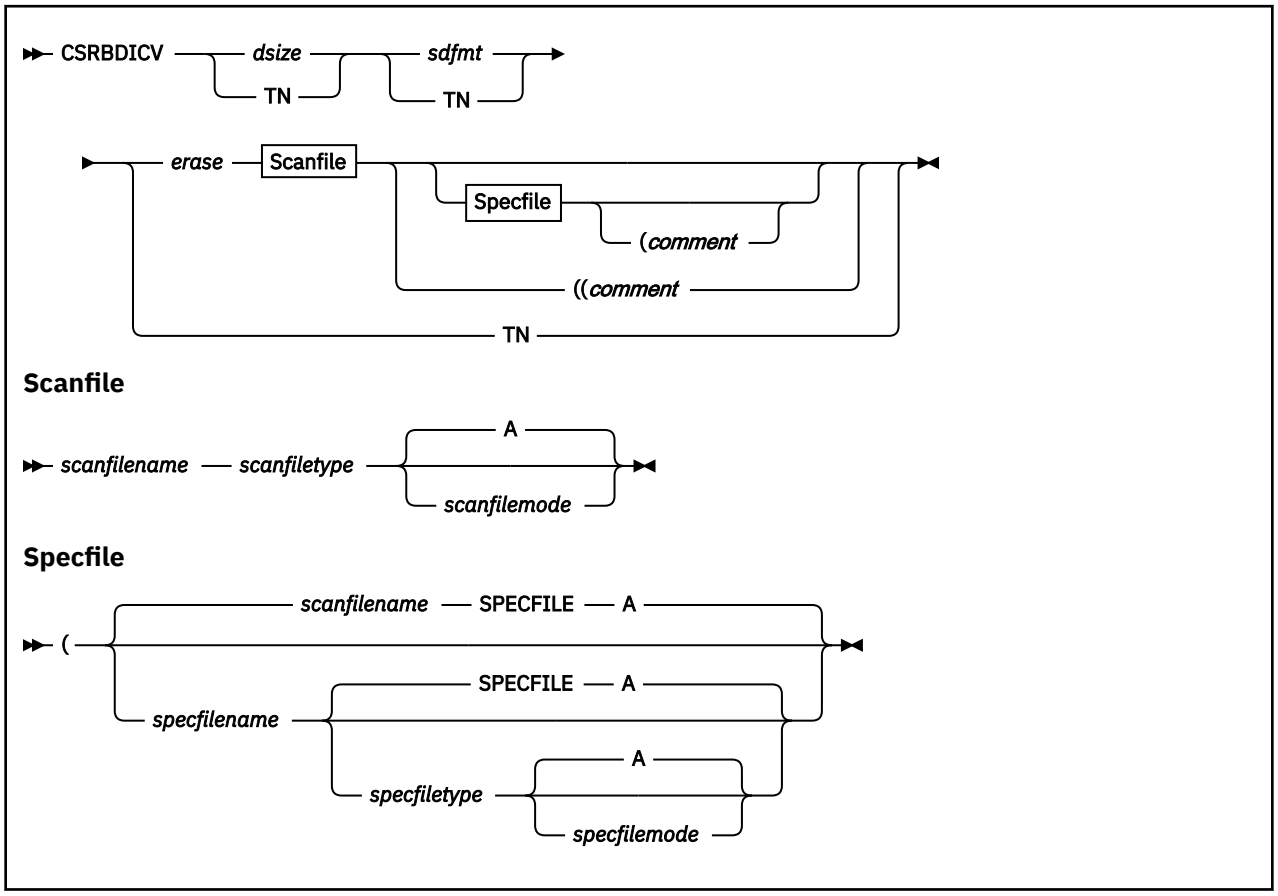
Messages and Return Codes

- DMS065E *option* option specified twice [RC=24]
- DMS066E *opt1* and *opt2* are conflicting options [RC=24]
- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS651E Append must be issued from CSLMAP [RC=40]
- DMS1088E Routine *rtnname* cannot be mapped because it is not loaded [RC=28]
- DMS1088E Routine *rtnname* cannot be mapped because it was not loaded with the specified attribute [RC=28]
- DMS1088E Routine *rtnname* cannot be mapped because it was not loaded with the specified group name [RC=28]
- DMS1088E Routine *rtnname* cannot be mapped because it was not loaded with the specified search criteria [RC=28]
- DMS1088E No routines can be mapped because none satisfy the specified search criteria [RC=28]
- DMS1088E Routine *rtnname* cannot be mapped because it was not loaded at the specified CSL path [RC=28]
- DMS1090E Invalid CSL path *path* specified [RC=24]
- DMS1110E CSLGEN encountered error executing *command*, rc=*retcode* [RC=40 from CSLGEN]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

CSRBDICV



Authorization

General User

Purpose

Use the CSRBDICV command to build tailored compression and expansion dictionaries based on a pattern scan of the actual data to be compressed. The results, for use with Data Compression Services, are:

- One abstract dictionary
- Two architectural format dictionaries

Operands

dsize
specifies, in kilobytes, the number of (8 byte) dictionary entries. The valid values are .5, 1, 2, 4, and 8.

sdfmt
specifies the format of the sibling descriptors in the dictionary. The valid values are 0 and 1.

Note: Data Compression Services (CSRCMPSC) only supports the use of format 1 sibling descriptors. However, when using the CMPSC hardware instruction directly (for example, not through the macro call), dictionaries may contain format 0 sibling descriptors.

erase
The valid values are:

X
causes an exit from CSRBDICV, if any of the dictionaries to be created already exist.

ED
causes any dictionaries to be created to be erased if they already exist.

EB
causes the dictionaries to be erased. It also causes *scanfilename* BDICT*sf* A to be erased even if the dictionaries already exist. In file type BDICT*sf*, *s* is the *dsize* and *f* is the *sdfmt*.

TN
returns an approximation of the maximum number of tree nodes, *maxnodes*, that can be built in the unallocated system storage currently available. This value can be used in the SPECFILE to allow the best tree structure to be built while scanning a source data file.

Note: For the small SCANFILE, the number of nodes that are built can be limited by the size of the SCANFILE itself, instead of the *maxnodes* value. For small data files, the SPECFILE *maxnodes* value should be set to approximately one-third of the actual number of nodes in the final tree size to improve the EXEC performance during the three scanning passes.

scanfilename scanfiletype
scanfilename scanfiletype scanfilemode
specifies the file to be scanned. The default file mode is A. The maximum length of a line in the scan file is 65535 bytes.

(*specfilename*
(*specfilename specfiletype*
(*specfilename specfiletype specfilemode*
specifies the file containing the scanning specifications. The default is *specfilename* SPECFILE A.

comment
can be anything. This can be used to identify the experiment (what is different this time). You can specify *comment* without specifying (*specfilename specfiletype specfilemode* by entering: ((*comment*

Usage Notes

1. The operands are positional. If you want to specify a particular operand, you must specify valid values for all the preceding operands, or if an operand is preceded by a "(", you can specify the left parenthesis and you will get the default value for that operand.
2. In file types BDICT*sf*, CEDICT*sf*, RSLTS*sf*, ACDICT*sf*, and AEDICT*sf*; *s* is the *dsize* and *f* is the *sdfmt*.
3. CSRBDICV scans an input file (*scanfilename scanfiletype scanfilemode*) by using the specifications from a control file (*specfilename specfiletype specfilemode*). It then builds an abstract compression and expansion dictionary called *scanfilename* CEDICT*sf* A. This dictionary can be used by the CSRCMPEV EXEC as input to test its effectiveness in compressing data.
4. Important statistics are written in *scanfilename* RSLTS*sf* A. The record of execution is written in BDICT LOG A. Progress reports, statistics, and displays are written in *scanfilename* BDICT*sf* A.
5. The maximum length of a line in any file is 65535 bytes.
6. For more information on the CSRBDICV command and how to use the spec files, see *z/VM: CMS Application Development Guide*.

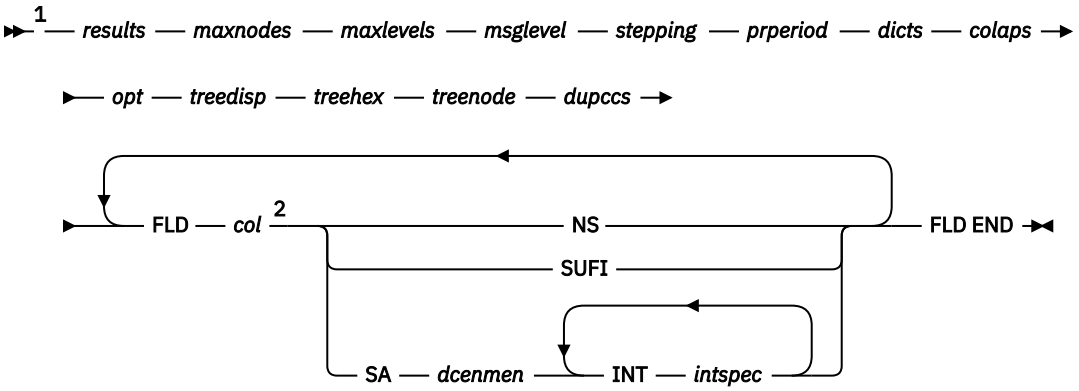
Spec File Information

This is an example of a spec file that you can use.

```
**The following is with a 4K-entry dictionary.
**Provides 30.88% compression (output/input) for the source of
**Chapter 5 of the ESA/390 Principles of Operation (30.32% if all output
**bits are concatenated together).
**Optimization (change x under opt to opt) improves compression by 0.7%.
**results maxnodes maxlevels msglevel stepping prperiod dicts
  r       40000    60        3        f 7 2 7  1000     af asm
**colaps opt treedisp treehex treenode dupccs
  aam     x   x        h       n        x
**FLD col type dcenmen              INT  intspec
```

```
FLD 1   sa   dce 4               INT  aeis 1 (40)
                                 INT  a12b3s (40)
FLD end
```

Note: The scanning specifications can be on one line or any number of lines. Before the specifications are parsed, the lines are concatenated with a blank inserted between each line and the following line. Lines beginning with ** are ignored (allows comments). The following is the format of a spec file:



Notes:

[1] Operands are positional.

[2] For the first FLD, *col* must be 1. For subsequent FLDs, *col* must be a whole number greater than the *col* of the previous FLD.

results
The valid values are:

X
causes no action

R
causes level-1 statistics to be written to the file *scanfilename* RSLTS*sf* A. The compression results from CSRCMPEV are written to the same file. You will have a permanent record of your experiments. R also causes CSRBDICV to make an entry in BDICT LOG A.

maxnodes
is a number specifying the maximum number of tree nodes to be built during scanning (15000-40000 is recommended). This number must be a whole number and equal to or greater than 512. For more information, see the description of the TN operand.

maxlevels
is a number specifying the maximum number of levels to be in the tree (at least 50 is recommended). This number must be a whole number and equal to or greater than one. Too many levels may cause the REXX control stack to become full. Regardless of the *maxlevels* specified and the number of levels actually in the tree, some of the statistic messages issued by the exec show a maximum of 125 levels and some show as few as 25 levels.

msglevel
specifies the types of error messages, progress reports, and statistics to be written to the screen during program execution, as follows:

Level
Written to Screen

0
Error messages (these indicate program termination)

1
Date, time, CSRBDICV arguments, contents of spec file, and important statistics

2
Important progress reports

3
Progress reports during scanning

4
More detailed progress reports during scanning, and characters of alphabet nodes during node processing

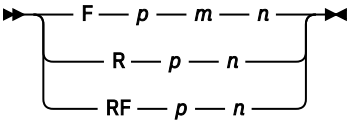
5
Parsed scanning specifications and statistics about children and additional extension characters

6
Statistics about duplicate child characters

Specification of a level causes that level and all lower-numbered levels to be written. Everything except the characters of alphabet nodes during node processing is always written in *scanfilename* BDICT*sf* A (except error messages for CSRBDICV arguments issued before the *scanfilename*, *s*, and *f* are known). Level-1 messages are optionally written in *scanfilename* RSLTS*sf* A.

Level 3 is recommended.

stepping



F *p m n*
F causes scanning of the first line of every *n* lines. When the entire file has been scanned, it is repositioned to its beginning, *m* lines are skipped, and the first line of every *n* lines is scanned. When the entire file has again been scanned, it is repositioned to its beginning, 2**m* lines (but not more than *n*-1 lines) are skipped, and the first line of every *n* lines is scanned. For example, F 7 2 7 causes scanning of the first line of every seven lines, then the third line of every seven lines, then the fifth line of every seven lines, and so forth (the seventh, second, fourth, and sixth). Scanning ends after the pass specified by *p*. *p*, *m*, and *n* must each be a whole number and equal to or greater than one. The program is (nodes, filesize, and line are intuitive):

```
do pass=1 to p
  ln=(1+m*(pass-1))
  ln=ln//n  /*// gives remainder*/
  if ln=0 then ln=n
  do ln=ln by n
 if nodes>maxnodes then leave pass
 if ln>filesize then leave
 scan line.ln
  end
  finis scanfile  /*Repositions to beginning*/
end
```

It has been observed that compression is not improved by making *p* larger than *n* (the same line should not be scanned more than once).

R *p n*
R causes scanning to begin at the line whose number is a random number between 1 and *n*. The number of the line to be scanned next is the number of the last line scanned plus a random number between 1 and *n*. When the entire file has been scanned, it is repositioned to its beginning, and the process is repeated. Scanning ends after the pass specified by *p*. *p* must be a whole number and equal to or greater than one. *n* must be a whole number and equal to or greater than two. The program is:

```
do pass=1 to p
  ln=0
  do forever
 if nodes>maxnodes then leave pass
 ln=ln+random(1,n)
 if ln>filesize then leave
 scan line.ln
  end
```

```
  finis scanfile
end
```

RF *p n*
RF causes a cursor to be set to 0 and then scanning to begin at the line whose number is the cursor plus a random number between 1 and *n*. The cursor is then advanced by *n*, and the line is scanned whose number is the cursor plus a random number between 1 and *n*, and so forth. When the entire file has been scanned, it is repositioned to its beginning, and the process is repeated. Scanning ends after the pass specified by *p*. Where:

p
must be a whole number and equal to or greater than one.

n
must be a whole number and equal to or greater than two.

The program is:

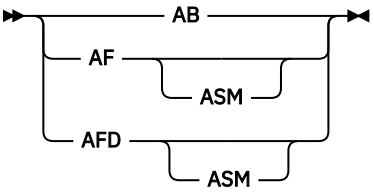
```
do pass=1 to p
  do cur=0 by n
 if nodes>maxnodes then leave pass
 ln=cur+random(1,n)
 if ln>filesize then leave
 scan line.ln
  end
  finis scanfile
end
```

F *p* 2 7 is recommended, where *p* depends on the size of your file and how long you want the optimization phase to take.

If you know from experience that *maxnodes* is large enough to allow scanning of the entire file, you can just as well use a stepping specification of F 1 1 1, which will simply cause scanning of each consecutive line in one pass (but the pass will still be divided into three program passes). The point of the stepping specification IS NOT to cause multiple passes over the scan file; it is to ensure there will be a sampling of the entire file (by reaching the end of the file at least once) when it is not known at what rate nodes will be built during scanning.

prperiod
is a number specifying the number of nodes to be built during the period covered by a progress report during scanning. This number must be a whole number and equal to or greater than one (1000 is recommended).

dicts



AB
causes an abstract dictionary (usable by CSRCMPEV EXEC) to be written as the file *scanfilename* CEDICT*sf* A.

AF
AF ASM
causes *scanfilename* CEDICT*sf* A to be written and also causes architectural-form dictionaries to be written as the files *scanfilename* ACDICT*sf* A and *scanfilename* AEDICT*sf* A. Each of the ACDICT*sf* and AEDICT*sf* files consists of a number of lines equal to the dictionary size. If ASM is omitted, each line is simply eight characters (many of which will be unprintable) that specify, by

means of their byte values, the contents of a doubleword dictionary entry. If ASM is specified, each line has the format of a Define Constant assembler statement, as follows:

```
DC X'hhhhhhhh'Entry nnnn |cccccccc| tag
```

Where:

hhhhhhhh
specifies the entry contents in hex.

nnnn
specifies the entry number in hex.

cccccccc
specifies the entry contents expressed as characters.

tag
If the entry is a character entry in the compression dictionary, *tag* is *CPTR=hhhh* if the entry has children or is null otherwise, where *hhhh* is the entry number of the first child in hex. If the entry is a character entry in the expansion dictionary, *tag* is *PPTR=hhhh* if the entry is a preceded entry or is null otherwise, where *hhhh* is the entry number of the preceding entry in hex. If the entry is a sibling descriptor in the compression dictionary or is the corresponding entry in the expansion dictionary, *tag* is *SD.* ASM also causes assembler statements to be placed before and after the dictionary entries as:

In *scanfilename* ACDICT*sf* A

```
ACDICT  CSECT      FORMED BY SCANNING scanfilename
                      scanfiletype scanfilemode
ACDICT  RMODE ANY  sK ENTRIES, FORMAT f
                      SIBLING DESCRIPTORS
(entries)
        END   ACDICT
```

In *scanfilename* AEDICT*sf* A

```
AEDICT  CSECT      FORMED BY SCANNING scanfilename
                      scanfiletype scanfilemode
AEDICT  RMODE ANY  sK ENTRIES, FORMAT f
                      SIBLING DESCRIPTORS
(entries)
        END   AEDICT
```

Prior to assembling the ACDICT*sf* or AEDICT*sf* file produced through the ASM option, you must rename the file to one which has a file type of ASSEMBLE.

AFD
AFD ASM
is like AF and also causes a readable combination of the ACDICT*sf* and AEDICT*sf* files to be written in *scanfilename* BDICT*sf* A. The presence or absence of ASM has the same effect as for AF.

colaps
The valid values are:

X
causes no nodes to be collapsed into their parents.

L
causes eligible leaf nodes to be collapsed into their parents after the scanning in the last program pass. A node is eligible if it has no sibling. Also, a node can contain at most 2-4 additional extension characters (AECs) and zero or one child character (CC) or zero or one AEC and multiple CCs.

AM
causes all eligible nodes to be collapsed into their parents after the scanning in the last program pass. Also, an additional extension character may be moved from a parent to a single child to make the parent able to be collapsed into its parent. A node is not changed if it begins with the *c* of any interruption specification, the last character of the parent is the same c, and the

number of characters represented by the parent is less than 11. The rule about c provides for good compression of strings, of length less than 11, of the same repeated character.

AAM
is like AM except (1) the collapsing (not the movement) is done after the scanning in all program passes, and (2) in other than the last program pass, a node is allowed to contain only one additional extension character (which makes it able to acquire multiple children during the scanning in the next program pass). AAM is recommended.

opt
The valid values are:

X
causes the optimization phase not to be executed. X is recommended for experiments in which field and interruption specifications are determined.

OPT
causes the optimization phase to be executed. OPT is recommended for production.

treedisp
Where the symbols cause some number of displays of the tree to be written in *scanfilename* BDICT*sf* A as shown below. The raw and collapsed trees will be written in program pass 3. The other displays will be written in program pass 3 if the optimization phase is not specified, or in the optimization phase if it is specified.

X
none

DR
raw

DRP
raw, then pruned

DRL
raw, then with children collapsed into parents

DL
children collapsed into parents

DS
siblings sorted by node values

DE
duplicate CC or SC nodes eliminated

DEC
eliminated, then with child counts placed in nodes

DC
child counts placed in nodes

DCP
child counts placed in nodes, then pruned

DP
pruned

D
all

treehex
The valid values are:

X
causes the tree display to contain hex translations only for nodes that are or are under unprintable alphabet nodes.

H
causes the tree display to contain hex translations for all nodes.

treenode
The valid values are:

X
causes the tree display not to contain node numbers.

N
causes the tree display to contain node numbers.

dupccs
The valid values are:

X
causes no action.

DCC
causes statistics about duplicate child characters to be written in *scanfilename* BDICT*sf* A.

FLD
denotes the start of a field specification.

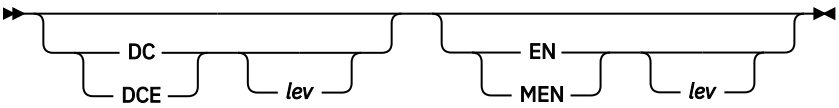
col
is the number of the column (character position) in which the field begins. *col* for the first field must be 1. *col* for each subsequent field must be a whole number and greater than *col* for the preceding field.

NS
causes the field not to be scanned.

SUFI
causes the field to be scanned until the first interruption caused by failure to find a match. One new node representing one extension character is then built, and the remainder of the field is not scanned.

SA
causes the entire field to be scanned. When an interruption occurs because of a failure to find a match or because the condition of an interruption specification is met, one or two new nodes are built (see *dcenmen*), and scanning resumes at the next character of the field after the one or more characters just placed in the new node or nodes.

dcenmen



DC
causes double character nodes to be built beginning at the level whose number is specified by *lev*. A double-character node contains one additional extension character.

DCE
is like DC except a double-character node is built only if the two characters in it are equal.

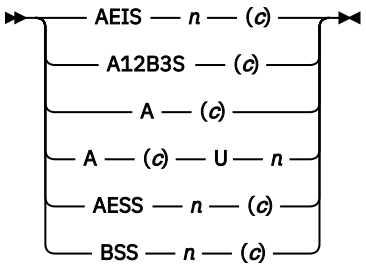
EN
causes one extra node (one extra node per scan interruption) to be built beginning when a normal node is built at the level whose number is specified by *lev*.

MEN
is like EN except that multiple extra nodes are built until an interruption specification is satisfied or the end of the current field is reached.

lev
must be a whole number and equal to or greater than two.

INT
denotes the start of an interruption specification.

intspec



AEIS *n* (*c*)
causes an interruption after the end of an initial string (a string beginning at the beginning of the string currently being scanned) of at least *n* instances of the character *c*. *n* must be a whole number and equal to or greater than one. *c* can be specified either as a single character or as two hex characters that are the EBCDIC code of the intended single character. If *c* is a blank or a lower-case character, it MUST be specified as two hex characters. (A blank would prevent proper parsing. The scanning specifications are converted to uppercase before parsing. These statements apply to all instances of (*c*).)

A12B3S (*c*)
causes an interruption after a subsequent set (a set not beginning at the beginning of the string currently being scanned) of only one or two instances of the character *c* but before a set of three or more instances of the character *c*. This is intended to get complete words followed by single blanks into the dictionary, and also a complete word followed by a period and two blanks. (ASCII might be used instead of EBCDIC, which is why *c* is a variable.) However, A12B3S will not cause an interruption if the string consists of one to three non-*c* characters followed by the *c* character, which is then followed by one to three non-*c* characters followed by the *c* character (thus allowing for a blank, or other break character, between two short words). This puts into the dictionary strings such as "may be," "is the," "can do," and "a bit."

A (*c*)
causes an interruption after encountering *c*.

A (*c*) U *n*
causes an interruption after encountering *c* unless *c* is the *n* character of the string currently being matched. *n* must be a whole number and equal to or greater than two.

AESS *n* (*c*)
causes an interruption after the end of a subsequent string (a string not beginning at the beginning of the string currently being scanned) of at least *n* instances of the character *c*. *n* must be a whole number and equal to or greater than one. (If an *n* of one is desired, A (*c*) may be able to be used, and it gives better scanning performance.)

BSS *n* (*c*)
causes an interruption just before a subsequent string of at least *n* instances of the character *c*. *n* must be a whole number and equal to or greater than one.

To obtain the best performance during scanning, multiple interruption specifications for the same field should be specified in the order in which they are shown and described above.

If a string as specified by AEIS is ongoing, no test is made of whether another interruption specification is satisfied, provided the AEIS specification precedes the other specifications.

END
denotes the end of the scanning specifications.

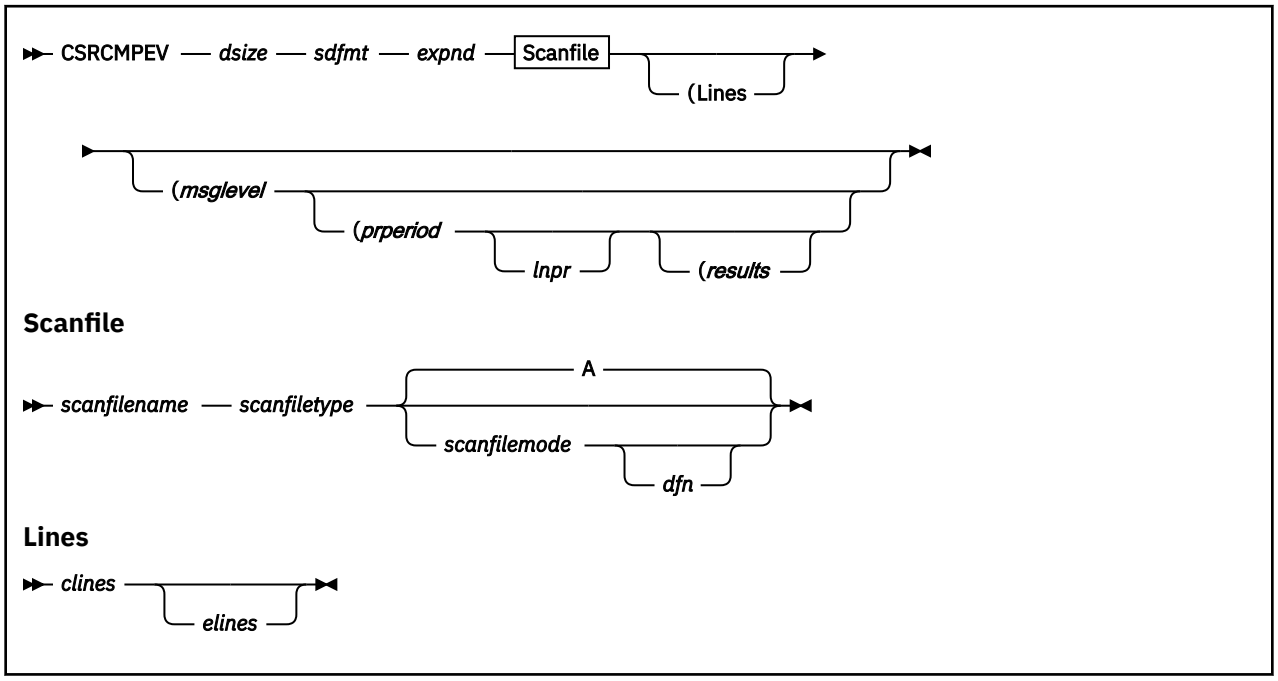
Messages and Return Codes

Messages are returned to the console and log files during execution. The level of message detail is controlled by the specification file "MSGLEVEL" parameter.

For an example of the dictionary builds, see *z/VM: CMS Application Development Guide*.

Error messages from this exec refer to argument errors in the exec or spec file arguments. Missing or wrong *file name file type file mode* errors may also be given for the necessary input files.

CSRCMPEV



Authorization

General User

Purpose

Use the CSRCMPEV command to compress and expand data using tailored dictionaries created by the CSRBDICV EXEC. Compression result reports are created automatically.

For more information, see Usage Note “2” on page 146.

Operands

dsize
specifies, in kilobytes, the number of (8 byte) dictionary entries. The valid values are .5, 1, 2, 4, and 8.

sdfmt
specifies the format of the sibling descriptors in the dictionary. The valid values are 0 and 1.

Note: Data Compression Services (CSRCMPSC) only supports the use of format 1 sibling descriptors. However, when using the CMPSC hardware instruction directly (for example, not through the macro call), dictionaries may contain format 0 sibling descriptors.

expnd
The valid values are:

H
causes the exec, when it writes the EXPND*sf* file, to insert a vertical bar (|) after the character string resulting from the expansion of each index symbol, to place a separating blank between each two output characters, and to put the hex values of the output characters, except for the vertical bars, above the output characters.

NH
(no hex) causes the exec to insert the vertical bars, but not the separating blanks or the hex values.

NHD
(no hex or delimiter) causes the output line to consist of only the expanded data, in which case the output line should be identical to the same line of the scan file. This is the only case in which the expanded data is compared to the contents of the scan file.

DB2
causes special formatting for a particular DB2® DATA file. Vertical bars are provided, hex is provided for only the first 32 characters of the line, and the rest of the line is written as a separate line.

scanfilename scanfiletype
scanfilename scanfiletype scanfilemode
scanfilename scanfiletype scanfilemode dfn
to specify the file (called the scan file) to be compressed and, optionally, a dictionary whose file name is different from that of the scan file. When *dfn* is not specified, *scanfilemode* defaults to A, and the dictionary is *scanfilename* CEDICT*sf* A. When *dfn* is specified, the dictionary is *dfn* CEDICT*sf* A. The maximum length of a line in any file is 65535 bytes.

lines
The valid values are:

clines
is the number of lines to be written in *scanfilename* CMPRS*sf* A. *clines* must be a whole number and greater than zero. The default is 100 or the number of lines in *scanfilename scanfiletype scanfilemode*, whichever is smaller.

elines
is the number of lines to be written in *scanfilename* EXPND*sf* A. *elines* must be a whole number and greater than zero. The default is 19 or the number of lines in *scanfilename* CMPRS*sf* A, whichever is smaller.

msglevel
The number specifies the types of error messages, progress reports, and statistics to be written to the screen during program execution, as follows:

Level
Written to Screen

0
Error messages (these indicate program termination, except for errors when comparing the EXPND*sf* file to the scan file).

1
Date, time, CSRCMPEV arguments, and important progress reports and statistics. For lines containing more than 10,000 characters, a progress report may be written for each 10,000 characters (see *lnpr*).

2
Other progress reports and the numbers of index symbols representing each length of character symbol.

3
Unused dictionary entries.

Specification of a level causes that level and all lower-numbered levels to be written. Everything except for progress reports about lines containing more than 10000 characters is always written in *scanfilename* BDICT*sf* A. Only a few level-1 messages are written in *scanfilename* RSLTS*sf* A.

The default is Level 1 and is recommended.

Note: The *msglevel* can be specified without specifying lines (*clines* and *elines*) by entering ((*msglevel*.

prperiod
is a number specifying the number of lines to be compressed or expanded during a progress-report period. *prperiod* must be a whole number and greater than zero. The default is 256.

Note: *prperiod* can be specified without specifying *lines* or *msglevel* by entering (((*prperiod*.

lnpr
The valid value is:

X
causes a progress report not to be written for each 10000 characters of a line.

results
The valid value is:

0
prevents statistics from being written.

1
causes statistics to be written in *scanfilename* RSLTS*sf* A, it also causes a record to be written in BDICT LOG A.

Results 1 is recommended and is the default.

Note: *results* can be specified without specifying *lines*, *msglevel*, or *prperiod* by entering ((((*results*.

Usage Notes

1. The operands are positional. If you want to specify a particular operand, you must specify valid values for all the preceding operands, or if an operand is preceded by a "(", you may specify the left parenthesis and you will get the default value for that operand.
2. In file types EXPND*sf*, CEDICT*sf*, CMPRS*sf*, BDICT*sf*, and RSLTS*sf*; *s* is the *dsize* and *f* is the *sdfmt*.
3. CSRCMPEV reads a file called the scan file (*scanfilename scanfiletype scanfilemode*) and compresses it using an abstract compression and expansion dictionary named *scanfilename* CEDICT*sf* A or *dfn* CEDICT*sf* A. CSRCMPEV writes the compressed data as the file *scanfilename* CMPRS*sf* A, and it then reads that file, expands the data, and writes the expanded data as the file *scanfilename* EXPND*sf* A.
4. During the expansion processing, CSRCMPEV optionally compares the expanded data to the contents of *scanfilename scanfiletype scanfilemode* and reports any differences (which will occur only if there is an error in the BDICT or CMPEXP exec). Although CSRCMPEV compresses all of *scanfilename scanfiletype scanfilemode*, in order to determine the amount of compression, it writes only a specified amount of compressed data in the CMPRS*sf* file. Then, although CSRCMPEV reads all the CMPRS*sf* file in order to expand the data and compare it to *scanfilename scanfiletype scanfilemode*, CSRCMPEV writes only a different specified amount of the data in the EXPND*sf* file (which is useful if EXPND*sf* might otherwise exhaust the capacity of the A disk). If the CMPRS*sf* or EXPND*sf* file already exists, it will be replaced.
5. Progress reports and statistics are optionally displayed on the screen and always written in *scanfilename* BDICT*sf* A. What is written to the screen can be controlled by means of the *msglevel* argument.
6. The most important statistics are written in *scanfilename* RSLTS*sf* A, and a record is written in BDICT LOG A. These actions can be prevented by changing the value of argument, RESULTS, for the exec from one (1) to zero (0).
7. The maximum length of a line in any file is 65535 bytes.

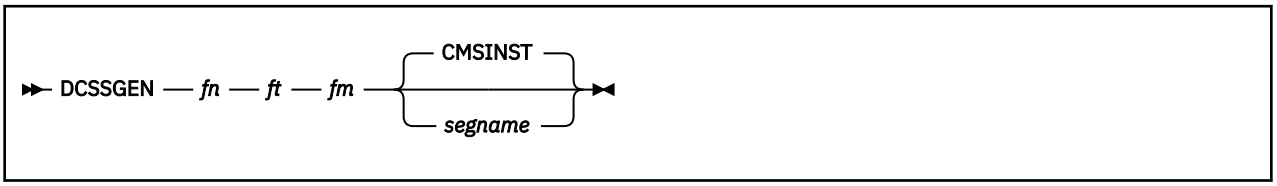
   Attention: If EXPND options 'H', 'NH', or 'DB2' are used to place hex data representations and/or '|' separation characters into the expanded output file, the resulting output lines must not exceed 65535 bytes. If they do, EXECIO errors will be raised while trying to write the output data. Use the 'NHD' EXPND option to write the expanded output without separation characters and hex representation at the exact size of the original file to avoid these EXECIO errors.
8. For more information on CSRCMPEV, see *z/VM: CMS Application Development Guide*.

Messages and Return Codes

Messages are returned to the console and log files during execution. The level of message detail is controlled by the specification file "MSGLEVEL" parameter.

Error messages from this exec refer to errors in the exec argument or errors in dictionary entries. If the control files are missing or have a wrong *file name*, *file type*, or *file mode*, errors could occur during compressing or expanding the input files.

DCSSGEN



Authorization

Saved Segment Administrator

Purpose

Use the DCSSGEN command to load and save a saved segment that contains frequently used execs and XEDIT macros. You can put the execs and XEDIT macros into a discontinuous saved segment (DCSS) or into a member of a segment space. When an exec or editor macro resides in a saved segment, many users can share the same executing copy.

DCSSGEN performs the following operations:

1. Processes a load list file, sequentially loading each exec and XEDIT macro into storage
2. Saves the saved segment
3. Writes a load map to the A-disk as *segname* DCSSMAP A

An alternative method for saving execs and XEDIT macros in shared storage is to use the SEGGEN command to load them into a logical saved segment. For more information on segment spaces, DCSSs, and logical saved segments, see *z/VM: Saved Segments Planning and Administration*.

Operands

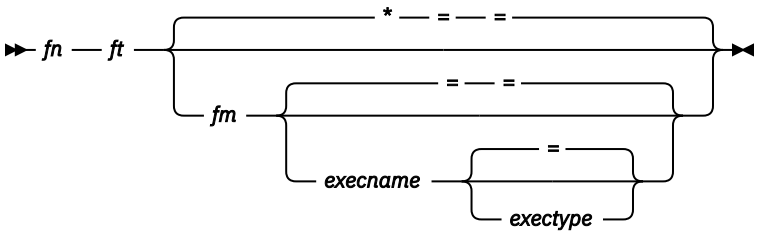
fn ft fm
is the file ID of your load list. For more information about the load list, see Usage Note "4" on page 148.

segname
is the name you want to assign to the saved segment. The default segment name is CMSINST.

Usage Notes

1. This command is no longer used during z/VM installation and service to build the CMS installation saved segment. The CMS installation saved segment (default name CMSINST) is built as a logical saved segment in the HELPINST physical saved segment. Do not use the DCSSGEN command to build a saved segment called CMSINST or any other name specified in the SYSPROF EXEC as the name of the installation saved segment. For more information about building the CMS installation saved segment, CMSINST, see *z/VM: Installation Guide*.
2. Before the DCSSGEN command can be used to load and save the saved segment, the CP DEFSEG command must be used to define the saved segment to CP.
3. To issue the DCSSGEN command, you must be authorized to perform the CP SAVESEG operation.
4. Before you enter the DCSSGEN command, you must create a file that contains a load list of the execs and XEDIT macro instructions to be loaded into the saved segment. **The load list must be a fixed-format file with a logical record length of 80.** Each record in the file must contain the file ID of one exec or XEDIT macro or a comment. DCSSGEN processes the records sequentially.

   The format of a DCSSGEN load list entry is:



Where:

fn
specifies the file name of the exec or editor macro to be loaded.

ft
specifies the file type of the exec or editor macro to be loaded.

fm
specifies the file mode of the exec or editor macro to be loaded. If the file mode is specified as *, DCSSGEN loads the first file in the disk search order that satisfies the file name and file type qualifications.

execname
specifies the file name to be assigned to the loaded exec or editor macro. The default is = (equals sign), which means the present file name is to be used.

exectype
specifies the file type to be assigned to the loaded exec or editor macro. The default is = (equals sign), which means the present file type is to be used.

The file name and file type of the exec or editor macro can each be from one to eight characters. The valid characters are:

A-Z
a-z
0-9
$ (dollar sign)
# (pound sign)
@ (at sign)
+ (plus sign)
- (hyphen)
: (colon)
_ (underscore)

The *execname* and *exectype* may also be from 1-8 characters; but they are not limited to the file name and file type character set. The only characters **not** valid within an *execname* and *exectype* are:

= (equals sign)
* (asterisk)
) (right parenthesis)
((left parenthesis)
X'FF'

To enter a comment in the load list, type an asterisk (*) in column one followed by the text of the comment.

For example, your load list entries may look like this sample for a saved segment named MYEXECS:

```
* Rename RDRLIST EXEC to MAIL EXEC
RDRLIST EXEC * MAIL =
FILELIST EXEC S
SYSPROF EXEC S
PARSE XEDIT S
DISCARD EXEC S
```

```
NOTE EXEC S
PROFNOTE XEDIT S
ALL XEDIT S
```

Before you process your load list, remove the comments and unnecessary blanks from the source program to conserve storage space. The EXECUPDT command with the NOCOMMENTS option removes all comments and leading blanks. One comment line containing the exec name and exec type is inserted at the beginning of the file. If the source file contains Double-Byte Character Set (DBCS) characters, also specify the ETMODE option. For more information, see "EXECUPDT" on page 256.

In the load map file, the records copied from your load list file are left-justified. The records created during the build process are indented five spaces. Comments are also copied from your load list file, with an asterisk (*) in column one followed by the text.

Examples

The load map file (MYEXECS DCSSMAP A) for the sample load list would look like this:

```
* RENAME RDRLIST EXEC TO MAIL EXEC
RDRLIST EXEC * MAIL =
  15:41:59 10/22/85 copy of RDRLIST  EXEC   S loaded as MAIL     EXEC
  EXISBLK - 280000   FBLOCK - 280100   LENGTH - 001C40
FILELIST EXEC S
  15:41:58 10/22/85 copy of FILELIST EXEC   S loaded as FILELIST EXEC
  EXISBLK - 280020   FBLOCK - 281D40   LENGTH - 0018C8
SYSPROF EXEC S
   7:30:18 11/26/85 copy of SYSPROF  EXEC   S loaded as SYSPROF  EXEC
  EXISBLK - 280040   FBLOCK - 283608   LENGTH - 002178
PARSE XEDIT S
   8:47:55 12/18/84 copy of PARSE    XEDIT  S loaded as PARSE    XEDIT
  EXISBLK - 280060   FBLOCK - 285780   LENGTH - 0024A0
DISCARD EXEC S
  15:40:32 10/22/85 copy of DISCARD  EXEC   S loaded as DISCARD  EXEC
  EXISBLK - 280080   FBLOCK - 287C20   LENGTH - 0012B0
NOTE EXEC S
  11:03:53 10/24/85 copy of NOTE     EXEC   S loaded as NOTE     EXEC
  EXISBLK - 2800A0   FBLOCK - 288ED0   LENGTH - 005310
PROFNOTE XEDIT S
  15:41:55 10/22/85 copy of PROFNOTE XEDIT  S loaded as PROFNOTE XEDIT
  EXISBLK - 2800C0   FBLOCK - 28E1E0   LENGTH - 000980
ALL XEDIT S
  15:41:05 10/22/85 copy of ALL      XEDIT  S loaded as ALL      XEDIT
  EXISBLK - 2800E0   FBLOCK - 28EB60   LENGTH - 001298
  *** End of Source List ***
  MYEXECS built at 15:56:34 on 12/02/86
```

Messages and Return Codes

- DMS284E The saved segment is not completely inside the virtual machine. You need 16MB.
- DMS288E *segname* not saved
- DMS288I *segname* not saved
- DMS298R An error has been detected while building the saved segment. Do you still want the saved segment saved? Enter 1 (YES) or 0 (NO).

DEBUG

Authorization

General User

Purpose

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

Usage Notes

1. CMS no longer supports the DEBUG subcommand environment. The functions provided by the old DEBUG subcommands are still available using CP DISPLAY, TRACE and PER.
2. In CMS, program and external interrupts no longer call DEBUG.
3. 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.
4. The DEBUG command is valid only if you enter it from the VM READ of CMS ABEND processing. (This is the VM READ 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.

Responses

This screen shows the information the DEBUG command displays:

```
DMS989I The state of the virtual machine at time of ABEND
follows:
Mode of virtual machine = XC   PSW = 00C02000 82FCDE72
GPR  0 = FFFFFFFF  0000D800  00000008  0000D808
GPR  4 = 82818495  4085A6A2  000000FF  00000000
GPR  8 = E2859584  40948540  A3964083  8193864B
GPR 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
AR   0 = 00000000 00000000 00000002 00000000
AR   4 = 00000000 00000000 00000002 00000000
AR   8 = 00000000 00000000 00000002 00000000
AR  12 = 00000000 00000000 00000002 00000000
```

Figure 3. DEBUG Output

Note: For an XA mode virtual machine, the last four lines for access registers will not be displayed.

Messages and Return Codes

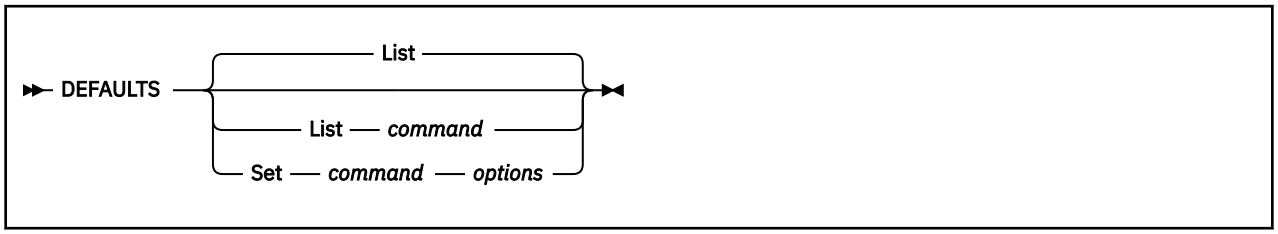
- DMS906E DEBUG command not valid at this time.

- DMS989I The state of the virtual machine at time of ABEND follows:

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

DEFAULTS



Authorization

General User

Purpose

Use the DEFAULTS command to set up default options or display the current default options for the commands shown in Table 11 on page 153.

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.

Operands

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

List
specifies the current default options for the command indicated are to be displayed. If no command is specified, all the commands listed in Table 11 on page 153 and the current default options are displayed. This is the default.

command
is one of the commands listed in Table 11 on page 153.

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 in Table 11 on page 153. The valid abbreviations for 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.

Table 11. Commands and Options that Can be Used with DEFAULTS

Command Name	Options
ALialist	`Profile fn` `Profile PROFALIA`
AUthlist	`Profile fn` `Profile PROFAUTH`
CERTmgr	`DATAbase dbpath`

Table 11. Commands and Options that Can be Used with DEFAULTS (continued)

Command Name	Options
CSLList	Profile *fn* Profile PROFCLST
CSLMap	Profile *fn* Profile PROFCSLM
Dirlist	Profile *fn* ALL Dirlist fn MDIsk Profile PROFDLST ACCessed NODirlist NOMdisk FILecontrol DIRControl NICk OWNer NONick READOnly READWrite
DISK Load	NOReplace MINPrompt FIFO Replace NOPrompt LIFO FULLPrompt NOType STACK TYPe
Filelist	Profile *fn* Filelist STAts ALLfile MIXEDON Profile PROFFLST Nofilelist SHAre AUThfile MIXEDOFF SEArch ALLDates PROFILE2 *fn2* PROFILE2 PROFFSHR SHOrtdate OWNer FULldate PROFILE3 *fn3* READOnly ISOdate PROFILE3 PROFFSEA READWrite VMDate PROFILE4 *fn4* PROFILE4 PROFFDAT
Help	Brief All DETail DEScript FORMat Parms OPTions NOTes ERRors Screen NOScreen
INCLUDE	MAP NOHOBSET FULLMap HOBSET NOMAP HOBSETSD
LOAD	MAP NOHOBSET FULLMap HOBSET NOMAP HOBSETSD
Maclist|MList	Profile *fn* Compact Profile PROFMLST Nocompact Compress Nocompress
NAMES	ALTMail COMdir PANel *pn* SERver VMLink MAIl

Table 11. Commands and Options that Can be Used with DEFAULTS (continued)

Command Name	Options
NETDATA RECEIVE	Log NOLog MINPrompt NOPrompt FULLPrompt Olddate NEwdate NOTebook *fn* NOTebook ALL NOTebook * SHORTDATE FULLDate ISOdate VMDate
NETDATA SEND	Log NOLog Type NOType NOAck Ack CLass A CLass c SHOrtdate FULLDate ISOdate VMDate
Note	Profile *fn* Profile PROFNOTE Short LONg NOSUBJect SUBJect LOG NOLog NOAck Ack NOTebook *fn* NOTebook ALL NOTebook * NONoTebook CLass A CLass c
Peek	Profile *fn* Profile PROFPEEK FRom recno FRom 1 FOr numrec FOr 25000 FOr *
RDrlist|RList	Profile *fn* Profile PROFRLST SHOrtdate FULldate ISOdate VMDate
READcard	NOReplace Replace MINPrompt NOPrompt FULLPrompt FIFO LIFO NOType STACK TYPe
Receive	Log NOLog Olddate NEwdate NOTebook *fn* NOTebook ALL NOTebook * Fullprompt Minprompt NOPrompt NOKeepcc Keepcc
Sendfile|Sfile	New Old NOType Type NOFilelist Filelist Log NOLog NOAck Ack CLass A CLass c RSCS SMTP UFTSYNC MIME UFTASYNC EBCDIC ASCII TEXT ASCII-INLINE ASCII-ATTACH BINARY BINARY-ATTACH BINARY-INLINE NETDATA VARREC NOSUBJect AUTOSUBJect
Tell	Msgcmd Message|Msgcmd MSG Msgcmd MSGNOH Msgcmd SMSG Msgcmd Warning|Msgcmd WNG

Table 11. Commands and Options that Can be Used with DEFAULTS (continued)

Command Name	Options
VMLink	NOKeep TYPe NOMODE0 Save Profile PROFVMLK Keep NOType MODE0 NOSave Profile *fn*

Usage Notes

1. The DEFAULTS command uses the GLOBALV command, which maintains a LASTING GLOBALV file on your disk or directory accessed as A. This file contains the options specified in a DEFAULTS command. However, *do not edit the LASTING GLOBALV file* to change the options. Use the DEFAULTS command, instead. For more information, see "GLOBALV" on page 366.
2. The FILELIST command has four profiles which set up the four FILELIST screens. By default, PROFILE PROFFLST is used for the STATS screen, PROFILE2 PROFFSHR is used for the SHARE screen, PROFILE3 PROFFSEA is used for the SEARCH screen, and PROFILE4 PROFFDAT is used for the ALLDATES screen.

   You can use the DEFAULTS command to change these four 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 three profiles as the default system profiles, issue:

   ```
   defaults set filelist profile2 myprof
   ```
3. You can use the CMS DEFAULTS SET command to change the CP command the CMS TELL command uses to send messages. If you set the CMS TELL command to use MSGNOH, you must be an authorized class B user for it to work.
4. For the DISK LOAD and READCARD commands, it is not possible to set the default option to either STACK FIFO or STACK LIFO. If STACK FIFO is the desired default, set the default option to either STACK or FIFO. If STACK LIFO is the desired default, set the default option to LIFO.
5. If you want to issue DEFAULTS from an EXEC program, you should precede it with the EXEC command; that is, issue:

   ```
   exec defaults
   ```

Examples

1. To display a list of your default options, issue:

   ```
   defaults
   or
   defaults list
   ```

   The response will look something like this:

   ```
   The following default options have been set:
     Alialist        options = PROFILE PROFALIA
     Authlist        options = PROFILE PROFAUTH
     Certmgr         options = DATABASE /etc/gskadm/Database.kdb
     ⋮
   To change any default options enter DEFAULTS Set Cmdname Opt1 <Opt2..>
   ```
2. To display a list of your default options for the CERTMGR command, issue:

   ```
   defaults list certmgr
   ```

   The response will look something like this:

   ```
   The following is a list of your default options for the CERTMGR command:
   defs DATABASE /etc/gskadm/Database.kdb
              DATABASE /etc/gskadm/Database.kdb
   To change these default options enter DEFAULTS SET CERTMGR Opt1 <Opt2..>
   ```

3. To set the CERTMGR command default database path as /etc/gskadm/Database.kdb, issue:

```
defaults set certmgr database /etc/gskadm/Database.kdb
```

4. To change the SENDFILE command default from NEW to OLD, issue:

```
defaults set sendfile old
```

Responses

1. DEFAULTS or DEFAULTS LIST:

```
The following default options have been set:
  command1        options = option1 option2 ...
  ⋮
To change any default options enter DEFAULTS Set Cmdname Opt1 <Opt2..>
```

2. DEFAULTS LIST *command*:

```


The following is a list of your default options for the command command:
defs option1 option2 ...
           option1 option2 ...
To change these default options enter DEFAULTS SET command Opt1 <Opt2..>
```

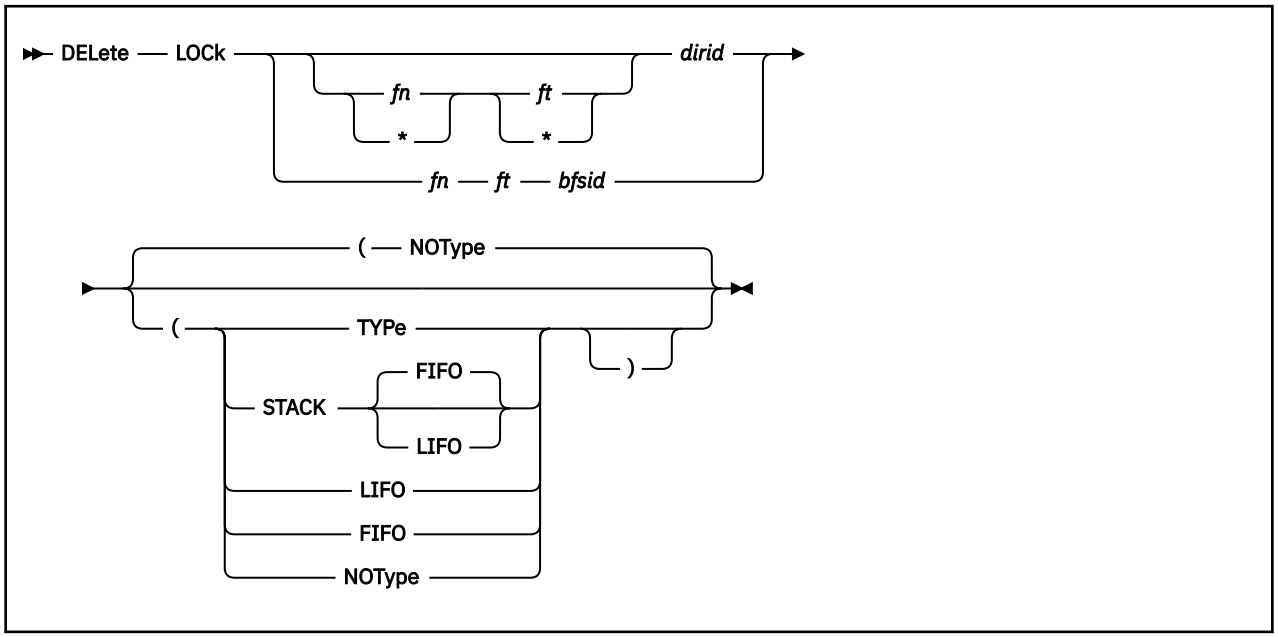
Messages and Return Codes

- DMS003E Invalid option: *option* [RC=24]
- DMS391E Unexpected operand(s): *operand* [RC=24]
- DMS641E No options specified [RC=24]
- DMS653E Error executing GLOBALV, RC=*nn* [RC=40]

This command might issue additional system messages. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

DELETE LOCK



Authorization

General User. A file pool administrator may enter this command for byte file system (BFS) files.

Purpose

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

Note: A different version of this command is available for users with file pool administration authority. For more information, see *z/VM: CMS File Pool Planning, Administration, and Operation*.

Operands

fn ft
specifies the name of the file to be unlocked. Special characters (* and %) can be used to designate a set of files. For more information on these special characters, see "Using Pattern Matching to Specify Sets of Files" on page 14.

dirid
specifies the directory name from which you choose to release the lock. If *fn* and *ft* are specified, this is the directory which contains the file to be unlocked. If *fn* and *ft* are not specified, this is the name of the directory which is to be unlocked. You can also use a file mode for the *dirid* if the directory is already accessed. If further detail is needed on *dirid*, see "Naming Shared File System (SFS) Directories" on page 6.

bfsid
identifies the byte file system (BFS).

For a BFS file, *fn* and *ft* are system-generated values that together uniquely identify the file within the BFS.

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. The default is NOType.

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

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

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

Usage Notes

1. For SFS files, if special characters are used for *fn* or *ft*, locks are not deleted in the directory for:
   - Subdirectories
   - Erased or revoked aliases
   - Files that you are not authorized to read or write
   - External objects

   Processing continues for the remaining file names that match the specified pattern.
2. 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.
3. If the DELETE LOCK command is issued from an exec or assembler program for a file pool that is active on the specified work unit, the command will fail.
4. 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 REXX exec with ADDRESS COMMAND in effect
5. DELETE LOCK may be used to delete an explicit lock on a byte file system file. However, the CMS short file name format of the file name must be used. You may not enter the DELETE LOCK command using the fully qualified byte file system (BFS) path name.

Messages and Return Codes

- DMS002E File *fn ft fm* | *dirname* not found [RC=28]
- DMS065E FROM option specified twice [RC=24]
- DMS066E *option1* and *option2* are conflicting options [RC=24]
- DMS069E Filemode *mode* not accessed [RC=36]
- DMS149E Userid *userid* not valid [RC=32]
- DMS637E Missing nodeid for the AT operand [RC=24]
- DMS647E Userid not specified for *nickname* in *userid* NAMES file [RC=32]
- DMS647E Localid not specified for *userid* at *node* in *userid* NAMES file [RC=32]
- DMS653E Error executing *command* rc=*nn* [RC=40]
- DMS1132E Invalid number of operands [RC=24]

- DMS1187E Too many subdirectory levels in *dirname* [RC=24]
- DMS1188E Filemode *mode* is not associated with a directory [RC=74]
- DMS1189E Filemode *mode* is associated with a top directory [RC=24]
- DMS1223E There is no default file pool currently defined [RC=40]
- DMS1139E You are not authorized to issue this command [RC=76]
- DMS1163E The DELETE LOCK command failed for *fn ft fm* | *dirname* [RC=*nn*]
- DMS1184E Directory *dirname* not found or you are not authorized for it [RC=28]
- DMS1184E File *fn ft* or directory *dirname* not found or you are not authorized for it [RC=28]
- DMS1206W There are no locks for *fn ft fm* | *dirname* [RC=4]
- DMS1209E Nickname *nickname* resolved to more than one userid; lock(s) can be deleted from only one userid at a time [RC=88]
- DMS1291E There are no unused work units available. [RC=88]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411
Errors in the Shared File System	“File Pool Server Messages” on page 1414

DELETE NAMEDEF

Authorization

General User

Purpose

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

Operands

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

Usage Notes

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 REXX exec with ADDRESS COMMAND in effect

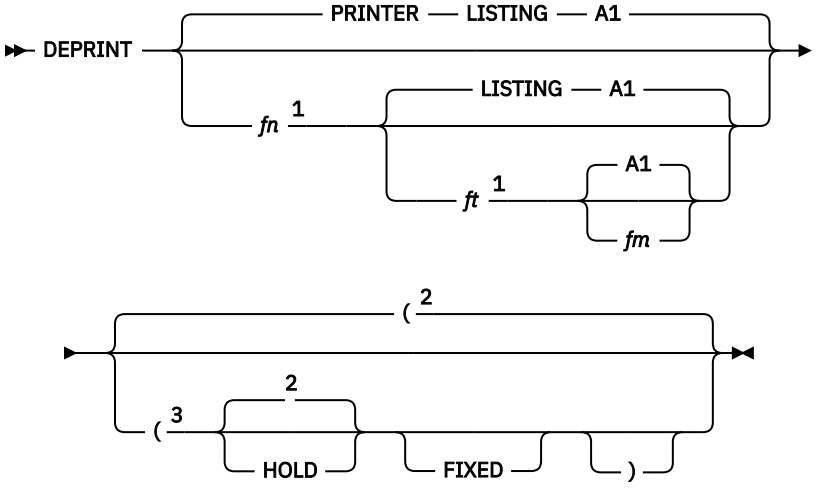
Messages and Return Codes

- DMS1192E Namedef *namedef* not found [RC=28]
- DMS1193E There are no namedefs to be deleted [RC=28]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

DEPRINT



Notes:

[1] Input over 8 characters will be truncated to 8 characters.
[2] The default is to purge the file.
[3] You can enter Options in any order between the parentheses.

Authorization

General User

Purpose

Use the DEPRINT command to take a printer file from your virtual reader and write it to a disk file without losing carriage control characters.

Operands

fn ft fm
specifies the file ID to be used for the output file. The default is PRINTER LISTING A1.

Attention:
The specified file ID must conform to the CMS file naming conventions. If the same file name, file type, and file mode already exists on your disk, the new file overwrites the original file.

HOLD
retains the reader spool file when the command completes. The default is to purge the input file. This option is assumed in the event of processing errors.

FIXED
writes the file to disk with a fixed record format. The default is to write the file with a variable record format. If the file is a VAFP print file and the FIXED option is chosen, the file created on disk will have an LRECL of 205.

Usage Notes

1. DEPRINT will not process reader files in HOLD status. DEPRINT chooses the first printer file in your virtual reader not in HOLD status and writes this file to a disk file or directory. If DEPRINT does not find any files it can process, you receive a return code 4 (No input print file).
2. DEPRINT does provide support for 3800 printer files, but it does not provide support for table reference characters (TRCs) in those files.
3. If a file is read in from the virtual reader by the DEPRINT command and the PRINT command is used to print the file, the printed file will contain an extra page with a line of unreadable characters.
4. DEPRINT processes class A spooled console output files (CON files).
5. If the reader file is a VAFP print file, the file created by DEPRINT will have a maximum lrecl of 205 (204 + 1 for carriage control).
6. In addition to the DEPRINT command, you can also use the preferred RECEIVE command with the KEEPCC option. See "RECEIVE" on page 806 for details.

Messages and Return Codes

- DMS002E File not found [RC=4]
- DMS062E Invalid * in fileid [RC=20]
- DMS069E Filemode *mode* is not accessed [RC=36]
- DMS070E Invalid parameter *parm* [RC=24]
- DMS105S Error *nn* writing file *fn ft fm* [on disk or directory] [RC=*xx*]
- DMS687E This file has a special format and cannot be received. [RC=10]
- DMS1215E File *fn ft fm* is locked by another user [RC=70]
- DMS1262S Error *nnn* opening file *fn ft fm* [RC=*xx*]
- DMS2166E Invalid channel command code found [RC=8]
- DMS2167I The tag for printer file number *spoolid* was: tag [RC=0]

DESBUF

Authorization

General User

Purpose

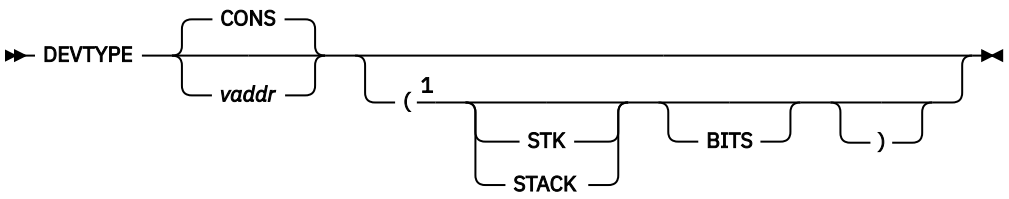
Use the DESBUF command to clear the console stack and any queued output lines. Any in-progress I/O (I/O that has been initiated, but has not yet completed) will not be affected.

Usage Notes

1. The console stack consists of the terminal input buffer and the program stack. For more information, see "Passing Commands and Data", in *z/VM: CMS Application Development Guide*.
2. An output line is queued if I/O is in progress at the time the output line is being processed. Output lines that are queued are always processed in a FIFO (First In/First Out) manner.
3. Enter the CONWAIT command before the DESBUF command if you want any in-progress I/O to complete and any queued output lines to be displayed before the DESBUF command is executed. If you do not enter the CONWAIT command, any queued output lines will be cleared; any in-progress I/O may complete after the DESBUF command is executed.

Note: Be careful when using the DESBUF command because clearing the entire program stack could remove data another program was relying upon.

DEVTYPE



Notes:

[1] You can enter Options in any order between the parentheses.

Authorization

General User

Purpose

Use the DEVTYPE command to display the device characteristics of any virtual address, with an option to stack the result. It also allows display of the data returned by CP DIAGNOSE codes X'24' or X'210'. See *z/VM: CP Programming Services* for details on these diagnose codes.

Operands

CONS
is the virtual console. This is the default.

vaddr
is the address of the device to be examined.

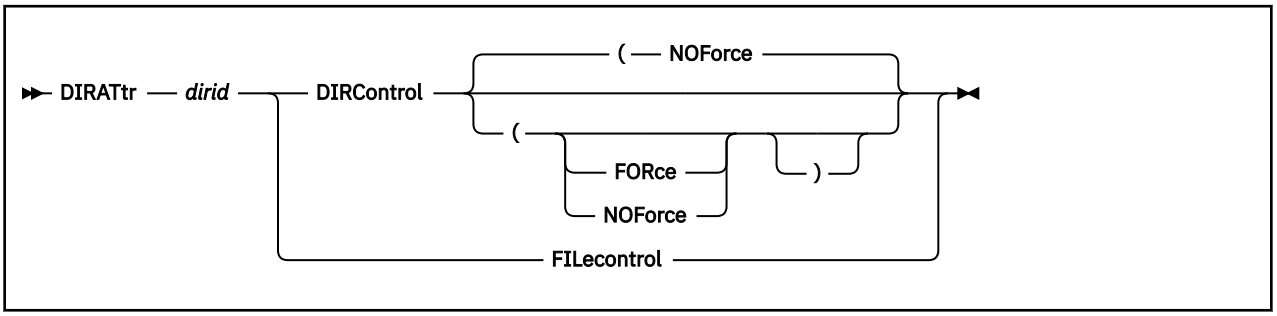
STK
STACK
causes the data to be stacked rather than displayed on the console.

BITS
displays (or stacks) the bits returned by CP DIAGNOSE codes X'24' or X'210'.

Messages and Return Codes

- DMS389E Invalid device address *devaddr* [RC=24]
- DMS2168I VDEV:TYPC= *class* TYPE= *type* STAT= *status* FLAG= *flag*
- DMS2169I RDEV:TYPC= *class* TYPE= *type* MDL= *model* FT/LN= *flag*

DIRATTR



Authorization

General User

Purpose

Use the DIRATTR command to set the directory attribute for an SFS directory.

Before setting the directory attribute, it is necessary to:

1. Release all explicit locks on the directory and its files.
2. Remove all aliases from the directory.
3. Revoke all authority to files in the directory and to the directory itself (including NEWREAD or NEWWRITE authority).
4. If the file pool is managed by DFSMS/VM, recall all migrated files in the directory. (You can use the DFSMS RECALL command to do this.)

You can erase the aliases and revoke the authorities by entering the DIRATTR command with the DIRCONTROL operand and the FORCE option. Explicit locks can be deleted by using the DELETE LOCK command.

Operands

dirid
is the identifier of the SFS directory for which you are setting an attribute. You can also use a file mode for the *dirid* if the directory is already accessed. For a more information on the directory ID, see "Naming Shared File System (SFS) Directories" on page 6.

DIRControl
sets the directory attribute to directory control. Directory control directories have characteristics that differ significantly from those of file control directories. For example, these directories are like minidisks and the performance can be better for read-only accessors. For more information, see "CREATE DIRECTORY" on page 97.

Files in directory control directories also cannot be placed in DFSMS/VM migrated status, but can be subject to automatic deletion by DFSMS/VM expiration processing.

FILecontrol
sets the directory attribute to file control. File control directories have file-level control characteristics. These include the ability to see committed changes made by other users when a file is closed and reopened, without reaccessing the directory; the ability to create aliases in the directory; and so on.

When FILECONTROL is specified, any existing directory control read and directory control write authorities are revoked. Any aliases associated with files in the directory are turned into dropped aliases.

Options

FORce
removes all existing authorities for the directory and its files. All aliases contained in the directory are erased and the directory is released. This operand may be used only when the DIRCONTROL operand is also specified.

NOForce
avoids the removal of existing authorities for the directory and its files. It also avoids the deletion of aliases in the directory, If, however, authorities or aliases exist, an error results. This operand may be used only when the DIRCONTROL operand is also specified. The default is NOFORCE.

Usage Notes

1. You can enter the DIRATTR command only for your own directories unless you have file pool administration authority.
2. In many cases, the DIRATTR command will fail if the specified directory is accessed or in use by you or any other user. When changing from file control to directory control, however, you can specify the FORCE option to make DIRATTR succeed, as long as there are no:
   - Open files in the directory
   - Explicit locks on the files or an explicit lock other than UPDATE on the directory

   You can change from directory control to file control when you have the directory accessed R/W or others have it accessed R/O, but not when you have the directory accessed R/O, or when someone else has it accessed R/W.

   If you have the directory accessed, it will be released. Other users who have the directory accessed will lose their access along with their authorities.
3. If you enter the command with the FILECONTROL operand, the directory is no longer eligible for use in a data space.
4. The DIRATTR command will fail if any user is writing to or reading from any file in the directory.
5. When the attribute is changed, any user who has the directory accessed will have it released automatically with no message issued.
6. If DIRATTR is issued from an exec or assembler program and the specified file pool is active on the current default work unit (that is, has some work on it that has not been committed or rolled back), the command will fail.
7. The DIRATTR command will fail if you are trying to change the attribute to DIRCONTROL and the directory contains any files in migrated status (that is, files whose data has been moved by DFSMS/VM into its storage repository.) You can use the DFSMS RECALL command to recall the files first. You can find out if DFSMS/VM has migrated any files by issuing:

   ```
   access dirname fm
   filelist * * fm (share
   ```

   and noting which files displayed contain an asterisk (*) in the "type" column of the screen. These files are in DFSMS/VM migrated status.
8. You can invoke the DIRATTR command from the command line, from an exec, or as a function from a program. No error messages are issued if DIRATTR 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 REXX exec with ADDRESS COMMAND in effect

Messages and Return Codes

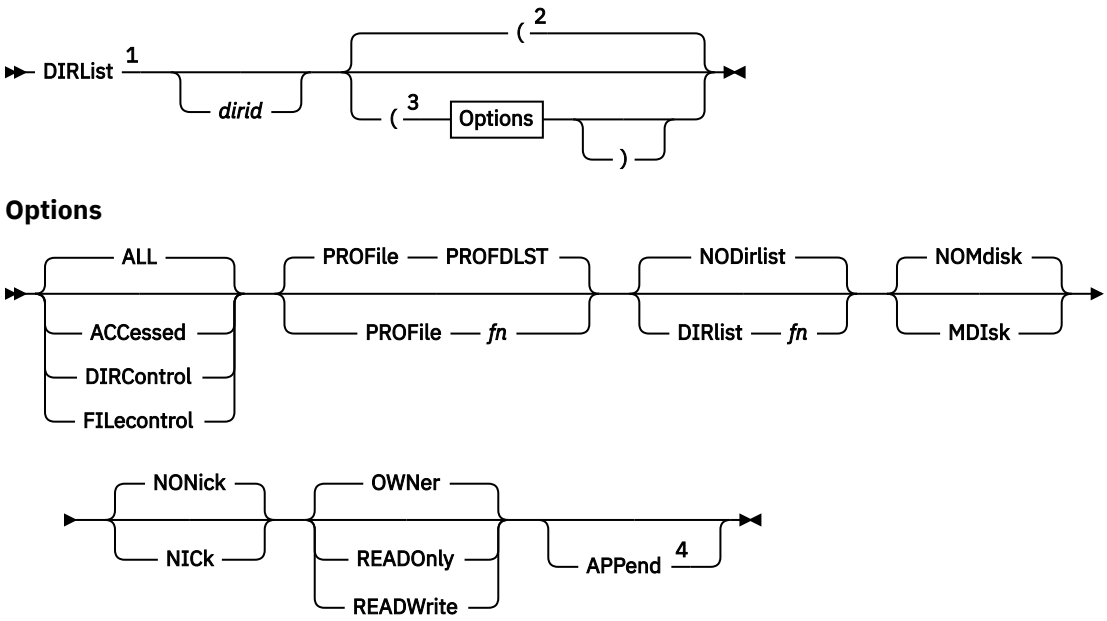
- DMS065E FORCE option specified twice. [RC=24]

- DMS065E NOFORCE option specified twice. [RC=24]
- DMS066E FORCE and NOFORCE are conflicting options. [RC=24]
- DMS066E NOFORCE and FORCE are conflicting options. [RC=24]
- DMS1184E Directory *dirid* is not found or you are not authorized for it. [RC=28]
- DMS2023E File pool *filepoolid* does not support the requested DIRATTR command [RC=88]
- DMS2032E FORCE option cannot be specified with FILECONTROL operand. [RC=24]
- DMS2032E NOFORCE option cannot be specified with FILECONTROL operand. [RC=24]
- DMS2033E Authorities exist on directory *dirid* and FORCE option was not specified. [RC=40]
- DMS2033E Aliases exist in directory *dirid* and FORCE option was not specified. [RC=40]
- DMS2034E An explicit lock is held on directory *dirid* or files in the directory. [RC=70]
- DMS2035W Directory attribute for directory *dirid* is already DIRCONTROL. [RC=4]
- DMS2035W Directory attribute for directory *dirid* is already FILECONTROL. [RC=4]
- DMS2036E Directory *dirid* contains a migrated file [RC=36]
- DMS2036E Directory *dirid* accessed or in use and FORCE option was not specified. [RC=36]
- DMS2036E Directory *dirid* contains a migrated file [RC=36]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411
Errors in the Shared File System	“File Pool Server Messages” on page 1414

DIRLIST



Notes:

[1] Special commands that can be used in the DIRLIST environment are AUTHLIST, DISCARD, and EXECUTE.
[2] Default options are shown above the line in the options group.
[3] You can enter Options in any order between the parentheses.
[4] APPEND is only valid when issued from the DIRLIST environment.

Authorization

General User

Purpose

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

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

Operands

dirid
specifies the SFS 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 some authority are listed. You can also use a file mode for the *dirid* if the directory is already accessed. If *dirid* is not specified, directories are listed beginning with your top directory. Also, if you use the NICK option, the user ID portion of the *dirid* can be a nickname that has been set up for a user or a group of users. For more information on how to specify *dirid*, see “Naming Shared File System (SFS) Directories” on page 6.

Options

ALL
lists all the directories in the specified directory structure for which you have some authority. Directories are listed regardless of their attributes. They are listed even if they are not accessed. The default is ALL.

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.

DIRControl
lists only directories with the directory control attribute. Directories are listed even if they are not accessed.

FILecontrol
lists only directories with the file control attribute. Directories are listed even if they are not accessed.

PROFile *fn*
specifies the name of an XEDIT macro to be executed when XEDIT is invoked by the DIRLIST command. If not specified, the default macro PROFDLST XEDIT is invoked. For more information, see Usage Note "10" on page 172.

DIRlist *fn*
specifies 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 more information, see Usage Note "5" on page 171.

NODirlist
specifies no list of directories is to be used. The default is NODIRLIST.

MDIsk
lists the minidisks **you** have linked. When the MDISK option is used with the ACCESSED option, only the minidisks you have accessed will be listed along with the directories that are listed. When the MDISK option is used with FILECONTROL, DIRCONTROL, or ALL, all the minidisks you have linked will be listed along with all the FILECONTROL, DIRCONTROL or ALL directories that are listed.

NOMdisk
does not list any minidisks you have linked. The default is NOMDISK.

NICk
allows the user ID portion of the *dirid*, passed in on a DIRLIST call, to be a nickname that has been set up for a user or group of users.

NONick
does not allow nicknames to be used as the user ID portion of the *dirid*. The default is NONICK.

OWNer
Specifies that if a temporary access of a directory is necessary, it will be done based on your ownership of the directory. This is the default. If the entry is a minidisk, this option is ignored.

READOnly
Specifies that if a temporary access of a directory is necessary, it will be accessed in R/O status regardless of your ownership or authority to it. If already accessed, the temporary access is not necessary, so this option will be ignored. If the entry is a minidisk, this option is ignored.

READWrite
Specifies that if a temporary access of a directory is necessary, it will be accessed in R/W status regardless of your ownership or authority to it. If the R/W attempt fails a R/O access attempt will be made. If already accessed, the temporary access is not necessary; so this option will be ignored. If the entry is a minidisk, this option is ignored.

APPend
lists the directories at the end of the existing list. APPEND is only valid within the DIRLIST environment.

Usage Notes

1. For more information on how to customize this command, see Appendix A, “Customizing Profiles for CMS Productivity Aids,” on page 1419.
2. 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 the directory or minidisk is not accessed).
   - The directory name or the minidisk address. Up to 70 characters are displayed on a line; therefore 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, and so forth.

   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.
3. Entering CMS Commands from DIRLIST

   Begin CMS commands entered from the command line with "CMS" to prevent XEDIT from decoding the command. This prevents CMS commands from being mistaken as XEDIT subcommands.
4. Using Special Commands from DIRLIST: EXECUTE, DISCARD, and AUTHLIST

   The EXECUTE command allows you to issue commands that use the directories and minidisks displayed by DIRLIST. The DISCARD command allows you to erase the directories and minidisks displayed by DIRLIST. The AUTHLIST command displays authority information about the directories displayed by DIRLIST. For more information, see Chapter 4, “Special Commands Used Within Other Commands,” on page 1385.
5. Saving a List of Directories, Minidisks or Both

   To save a list of displayed directories, minidisks, or both, 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, minidisks, or both, file it under a different file name. You have two choices on how to do this, issue FILE from the command line:

   - Specifying a different *filename*. For example, you could issue FILE MYDIRS.
   - And then use the RENAME command.

   Saving a list of directories, minidisks, or both, is useful when you want certain directories and minidisks 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 and minidisks previously saved.

   Note: Your entire saved list will always be displayed when you specify the DIRLIST *fn* option, even if you have also specified the ACCESSED, FILECONTROL or DIRCONTROL options, and some or all the files in your saved DIRLIST do not meet these criteria. Also if your saved list was from a DIRLIST that used the MDISK option your entire saved list will be displayed even if the MDISK option is not used when you specify the DIRLIST *fn* option. A "not found" message will appear beside each file not meeting the criteria you have specified.
6. Tailoring the DIRLIST Command Options

   The DEFAULTS command can be used to set up options and override command defaults for DIRLIST. The options 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. For more information, see "DEFAULTS" on page 153.

7. Issuing Commands from the DIRLIST Screen

   You can issue commands directly from the line that displays a directory, by typing the command in the command (Cmd) column. 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 are finished typing the command, erase the rest of the line by pressing Erase EOF, or space over the rest of the line. Press Enter to execute the command. You may use the DELETE key to erase the rest of the line, but do not use it to erase only part of the rest of the line. After the command executes, press the PF2 key to refresh your screen so an updated screen is displayed to type other commands. For more information, see "EXECUTE" on page 1393.

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

8. 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.

   For more information, see "FILELIST" on page 291.

9. Using Special Symbols

   These 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.

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

10. 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.
PF1	Help	Displays a help screen that gives the DIRLIST command description.

Key	Setting	Action
PF2	Refresh	Updates the list to indicate new directories, renamed directories, and so forth, using the same operands and options as when DIRLIST was invoked.
PF3	Quit	Exits from DIRLIST.
PF4	Sort(fm)	Sorts the list alphabetically by file mode, use the SMODE XEDIT macro.
PF5	Sort(dir)	Sorts the list alphabetically by directory name, use the SDIR XEDIT macro to do this.
PF6	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. **Note:** The PF 6 key is used with directories only, therefore, issuing it on a minidisk entry will cause an error. For more information, see "AUTHLIST" on page 1389.
PF7	Backward	Scrolls backward one screen.
PF8	Forward	Scrolls forward one screen.
PF9		Not assigned.
PF10		Not assigned.
PF11	Filelist	Issues the command FILELIST * * for the directory or minidisk where the cursor is placed. Displays a list of files for the designated directory or minidisk whether it is accessed or not accessed. If the directory or minidisk 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 back to its previous location.

Note: If you have 24 PF keys, PF keys 13 - 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 are not accessed are listed last, sorted alphabetically by directory name. SMODE is an XEDIT macro.

SDIR
Sorts the list alphabetically by directory name.

11. Using nicknames with DIRLIST

    If there is a user ID that is the same as a nickname, the nickname takes precedence when the NICK option is used. The nickname is used in the user ID portion of *dirid*.

    For example, you have a MYDEPT nickname which is a list of your department members. There is also a MYDEPT user ID. If you want to use the MYDEPT nickname to display a list of all your department member's GOODIES directories (and subdirectories) for which you are authorized, you would enter:

```
DIRLIST MYDEPT.GOODIES (NICK
```

If you want to use the MYDEPT user ID to display a list of user MYDEPT's GOODIES directories (and subdirectories) for which you are authorized, you would enter:

```
DIRLIST MYDEPT.GOODIES (NONICK
```

If you allow the NICK/NONICK option to default to NONICK, you could also display user MYDEPT's GOODIES directories (and subdirectories) for which you are authorized by entering:

```
DIRLIST MYDEPT.GOODIES
```

12. The DIRLIST option of the DIRLIST command uses an input DIRLIST file containing a list of directory names. These directory names may not include nicknames.
13. Displaying your authorized directories through an XEDIT macro

    This is a sample of the XEDIT macro you can choose to duplicate for a convenient way of displaying all the directories in your default file pool you have read or write access to.

```
/*********************************************************/
/* 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'
```

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

```
exec dirlist
```

Examples

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

====>
                                                       X E D I T  1  File
```

Figure 4. Sample DIRLIST Screen

For more information on each column, see "Responses" on page 175.

User SMITH enters,

```
dirlist (mdisk
```

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/Minidisk Address
      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
      C  191
      Y  19E
      S  200
      -  555


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

====>
                                                       X E D I T  1  File
```

Figure 5. Sample DIRLIST Screen with MDISK Option

For more information on each column, see "Responses" on page 175.

For more examples using the DIRLIST command, see *z/VM: CMS User's Guide*.

Responses

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

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

Where:

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

Fm
specifies the file mode letter of the accessed directory

- Dash
specifies the directory is **not** accessed

Directory Name
specifies 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:

*****
specifies the command was executed successfully (RC=0).

***n**
specifies the number for the return code (RC=n).

***?**
specifies the command was unknown to CP or CMS (RC=-3).

***!**
specifies the command is not valid in CMS subset. For a list of commands valid in the CMS subset mode, see *z/VM: CMS User's Guide*.

Issuing the DIRLIST command with the MDISK option displays this information:

```
Cmd Fm  Directory Name/Minidisk Address
    fm  directory name
    fm  minidisk address
    .   .
    .   .
```

Where:

Cmd
specifies a command column for typing and entering commands for the directory or minidisk displayed on the line. You may use special characters. For example, / and = to represent what is already displayed on the line.

Fm
specifies the file mode letter of the accessed directory or minidisk. The - (dash) means the directory or minidisk is **not** accessed.

Directory Name or Directory Name/Minidisk Address
specifies the complete directory name or the minidisk address.

Messages and Return Codes

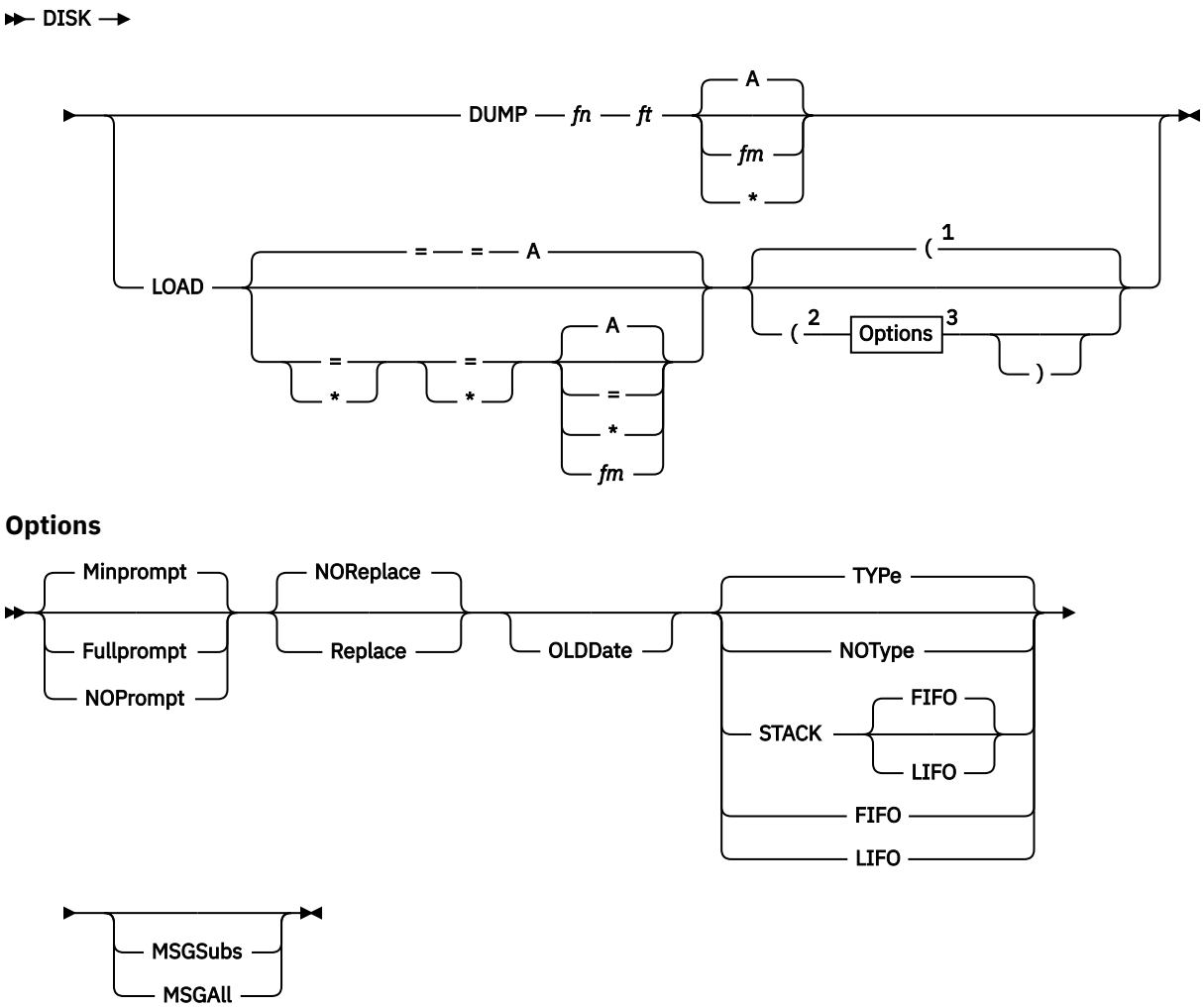
- DMS002E File *fn ft fm* not found [RC=28]
- DMS069E Filemode *mode* not accessed [RC=36]
- DMS105S Error *nn* writing file to XEDIT [RC=100]
- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS557E No more storage to insert lines [RC=4]
- DMS651E APPEND|SMODE must be issued from DIRLIST [RC=40]
- DMS651E X$NDIR$X must be issued from FILELIST or DIRLIST [RC=40]
- DMS653E Error executing LISTDIR, rc=*nn* [RC=*nn*]
- DMS1184E Directory *dirname* not found or you are not authorized for it [RC=28]
- DMS1187E Too many subdirectory levels in *dirname* [RC=24]

- DMS1188E Filemode *mode* is not associated with a directory [RC=74]
- DMS1189E Filemode *mode* is associated with a top directory [RC=24]
- DMS1223E There is no default file pool currently defined [RC=40]
- DMS1210E Directory *dirname* not found [RC=28]
- DMS1227E No filemode is available to access directory|minidisk [RC=0]
- DMS1228E Error executing access for directory|minidisk, rc=*nn* [RC=0]
- DMS1229E Directory is empty [RC=0]
- DMS1234E Error executing FILELIST, rc=*nn* [RC=*nn*]
- DMS1249I Directory has been temporarily accessed (read/only) as filemode *fm* [RC=0]
- DMS3995W You are not authorized to access in read/write mode [RC=4]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411
Errors in the Shared File System	“File Pool Server Messages” on page 1414

DISK



Notes:

[1] Default options are shown above the line in the options group.
[2] You can enter Options in any order between the parentheses.
[3] If multiple options from a single option stack are entered on the command, the last option specified will override the previous options.

Authorization

General User

Purpose

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. For more information, see Usage Note “5” on page 180.

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 *= = fm*
loads a file or files from the spooled card reader and writes them as CMS files on the specified disk or accessed SFS directory. The file name and file type are obtained from the card stream. If the file mode is specified as an asterisk '*' or equal sign '=', the file mode is obtained from the card stream. These may be changed through the use of the prompting facility. 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 even if there is a problem found in the sequence number check.

The DISK LOAD function checks for characters that are not valid in the file ID field of the reader file to be loaded. If a character that is not valid is found, the message DMS496S is printed at the console informing the user a not valid 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

DISK LOAD Options

Fullprompt
specifies a prompt is issued for each file in the spool file.

Minprompt
specifies 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. The default is MINPROMPT.

NOPrompt
specifies 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 a file is not received that would overlay an existing file on the receiving disk or directory. The default is NOREPLACE.

OLDDate
when specified, OLDDATE retains the date and time of the most recent update of the file before it is 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.

TYPe
specifies responses will be displayed to the terminal. For more information, see Response “1” on page 182.

NOType
specifies responses (excepting prompting messages) will not be displayed to the terminal.

FIFO
STACK
STACK FIFO
specifies responses will be placed in the stack in the order in which they would have been displayed. For more information, see Response “1” on page 182.

Note: STACK, STACK FIFO and FIFO are synonymous.

LIFO
STACK LIFO
specifies responses will be placed in the stack in the inverse order in which they would have been displayed. For more information, see Response "1" on page 182.

Note: LIFO and STACK LIFO are synonymous.

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. For more information on substitution data, see "Usage Notes" on page 180.

The lines are displayed or stacked in accordance with the FIFO, LIFO, NOTYPE, STACK or TYPE 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 FIFO, LIFO, NOTYPE, STACK or TYPE options.

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. For more information, see "RDR" on page 784 and "RECEIVE" on page 806.
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
   ```

   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:

   - Always use SENDFILE, which resets any continuous spooling options in effect.
   - Not spool the punch continuous.
3. You can read multiple files into separate files on disk or directory by spooling your reader continuous (CP SPOOL command with the CONT option) and issuing DISK LOAD with the NOPROMPT option.
4. You can use the DEFAULTS command to set up options and 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. For more information, see "DEFAULTS" on page 153.
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. If you specify the FULLPROMPT or MINPROMPT option the valid responses include, one of the:

   - Digits specified in the prompt

- Parenthetical words that follow 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 *fn1 ft1 fm1* (or *fn3 ft3 fm3*).

2 or Quit
Ends the command.

3 or Rename
Requests prompt message DMS1080R so the incoming file can be received using a different name.

7. If you receive prompt message DMS1081R 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 *fn ft fm* and repeats the original prompt message DMS1080R. 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.

8. 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.
9. If you specify the MSGSUBS or MSGALL option, the DISK LOAD command uses the following response format when returning the substitution data. This response has a fixed format and fixed length. The length is 69 characters.

   Note: Each individual field in the substitution line is initialized to an '*' followed by enough blanks to fill the field.

   The format is (each field is delimited by one blank):

   ```
   msgid fileid sentname oldfileid
   ```

   Where:

   msgid
   identifies the six character message identifier (four character message number and two character message format).

   fileid
   consists of three fields, each delimited by one blank:

   - file name (*fn*), length of eight
   - file type (*ft*), length of eight

- file mode (*fm*), length of two

This is the name of the file that was received.

sentname
consists of three fields, each delimited by one blank:

- file name (*fn*), length of eight
- file type (*ft*), length of eight
- file mode (*fm*), length of two

This is the name of the file as specified by the sender. This field will only be supplied if it differs from *fileid*.

oldfileid
consists of three fields, each delimited by one blank:

- file name (*fn*), length of eight
- file type (*ft*), length of eight
- file mode (*fm*), length of two

This is the name of the file replaced by *fileid*. This field will only be supplied if it differs from *fileid*. The only difference between the values of *fileid* and *oldfileid* will be the values of the *fm* fields.

Note: The MSGALL and MSGSUBS options only apply to those responses controlled by the FIFO, LIFO, NOTYPE, STACK and TYPE options.

10. If you try to load an empty file into a minidisk, you will get this message:

```
File fileid is empty; minidisk
does not support empty files
```

The file is not purged from your reader. You can DISCARD or PURGE the file from your reader, or try to receive it into an SFS directory that supports empty files. (It needs to be in a file pool managed by a server at CMS level 8 or later.) If the directory supports empty files, the record length, record format and file ID will be transferred to the new file. If the file pool does not support empty files, you will get this message:

```
File fileid is empty; filepool filepoolid
does not support empty files
```

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:

```
DMS1079R  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)

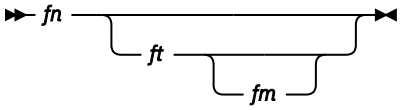
Receive fn1 ft1 fm1 and replace
fn2 ft2 fm2?
Reply 0 (NO), 1 (YES), 2 (QUIT), or 3 (RENAME)

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 you may specify when the name differs from the name of the incoming file.
- If you respond with a 3 (or RENAME) to prompt message DMS1079R, the following message appears and you must enter a file ID in the form




```
DMS1080R  Enter the new name for fn ft fm
```

- If you respond to prompt message DMS1080R with a file ID that names an existing file, you receive this prompt:

```
DMS1081R  Replace fn ft fm?
Reply 0 (NO), 1 (YES), or 2 (QUIT)
```

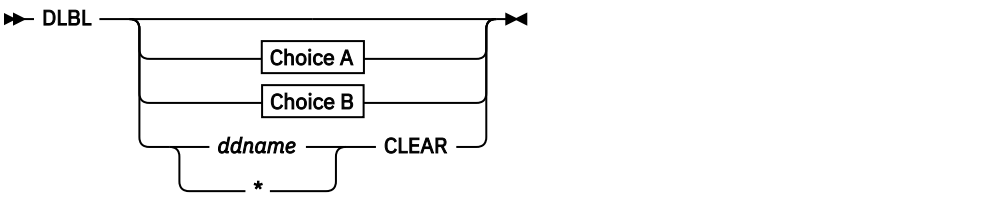
Messages and Return Codes

- DMS002E File *fn [ft [fm]]* not found
- DMS014E Invalid function *function*
- DMS024E File *fn [ft fm]* already exists; specify REPLACE option] [RC=28]
- DMS037E Filemode *mode*[(*vdev*)] is accessed as read/only
- DMS047E No function specified
- DMS054E Incomplete fileid specified
- DMS062E Invalid * in fileid
- DMS069E Filemode *mode*[(*vdev*)] not accessed
- DMS069E Output filemode *mode*[(*vdev*)] not accessed
- DMS070E Invalid parameter *parameter*
- DMS077E End card missing from input deck
- DMS078E Invalid card in input deck
- DMS078W Sequence error detected loading *fn ft*--expected *seqno1* found *seqno2*
- DMS104S Error *nn* reading file *fn ft fm* from disk or directory
- DMS105S Error *nn* writing file *fn ft fm* on disk or directory
- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS118S Error punching file
- DMS124S Error reading card file
- DMS205W Reader empty, reader not ready or empty reader file
- DMS257T Internal system error at address *address* (offset *offset*)
- DMS445W Invalid data in sequence field, bypassing sequence check
- DMS496S Invalid fileid *fn ft fm* found in input record [RC=100]
- DMS550W Date/Time data not present for file *fn ft*
- DMS639E Error in *routine* routine; return code was *nnnn*
- DMS671E Error loading file *fn ft fm*; rc=*nn* from RENAME
- DMS701W File *fileid* is empty; {minidisk|filepool *filepoolid*} does not support empty files [RC=74|88]
- DMS1123E Unknown response *text* ignored
- DMS1124W Spool file *spoolid* has been left in your reader because one or more files were not received [RC=1]
- DMS1138E Filesharing conflict [involving file *fn ft fm*] [RC=70]
- DMS1262S Error *nnn* opening file *fn ft fm* [RC=31 | 55 | 70 | 99 | 100]
- DMS1262S Error *nnn* closing file *fn ft fm* [RC=31 | 100]
- DMS1285S Default option *text* is invalid [RC=24]

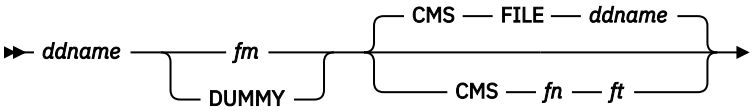
Additional system messages may be issued by this command. The reasons for these messages and their location are:

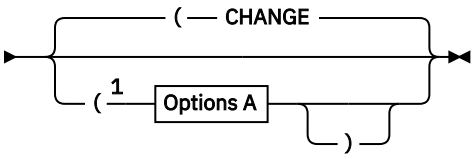
Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

DLBL

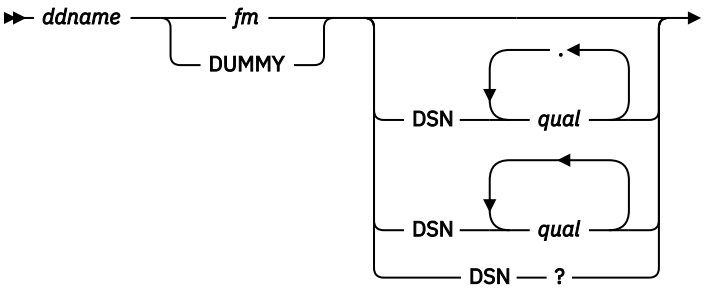


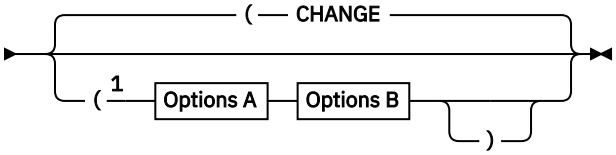
Choice A



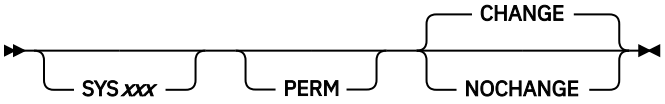


Choice B

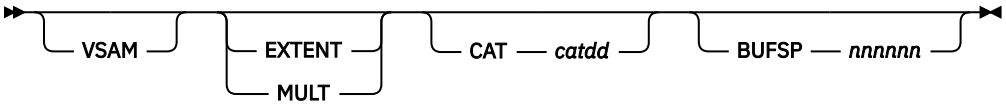




Options A



Options B



Notes:

[1] You can enter Options in any order between the parentheses.

Authorization

General User

Purpose

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. For more information on when a DLBL statement is required, see *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.

Operands

ddname
specifies a 1-7 character program ddname (OS) or file name (VSE), or dname (as specified in the FILE parameter of an access method services control statement).

*****
indicates all DLBL definitions, except those 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.

DUMMY
specifies 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. For more information, see Usage Note “3” on page 188.

CLEAR
removes any existing definitions for the specified ddname. Clearing a ddname before defining it ensures a file definition does not exist and that any options previously defined with that ddname no longer have any effect.

CMS *fn ft*
indicates 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. For more information on CMS/DOS users, see Usage Note “3” on page 188.

DSN
indicates this is a non-CMS file.

?
indicates 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.

qual
or
.qual
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 data set name, including periods between qualifiers. You can specify any number of names. For more information, see Usage Note “2” on page 188.

Options

SYS*xxx*
(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. For information on when the SYSxxx operand is required, see *VSE/VSAM Programmer's Reference*.

PERM
indicates 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 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 for the definitions to be merged. The default is CHANGE.

NOCHANGE
does not alter any existing DLBL definition for the specified ddname, but creates a definition if none existed.

VSAM
indicates 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 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 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. For more information on when EXTENT or MULT information is required, see *VSE/VSAM Programmer's Reference*.

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 the file definitions in effect, either of the following commands may be entered:

```
DLBL or QUERY DLBL
```

If, for example, you have previously entered:

```
assgn sys001 b
dlbl junk b dsn pdsf.l80.b80 ( sys001
```

and you then enter:

```
dlbl
```

the following information is displayed:

```
DDNAME  MODE LOGUNIT  TYPE CATALOG EXT VOL BUFSPC PERM DISK DATASET.NAME
JUNK     B1  SYS001   SEQ                          NO   DOS PDSF.L80.B80
```

For more information, see "Responses" on page 668.

If no DLBL definitions are in effect, the following message is displayed:

```
DMS324I  No user defined DLBL 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

DMS220R  Enter data set name:

test.input.source.d
```

If the data set name is entered interactively, the data set name *must* be entered in its exact form. If it is entered as a command, or from EXEC 2, the data set 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 by means of a CMS EXEC, the periods between qualifiers must be omitted, and the qualifiers must be 1-8 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 *qual* parameter. If you are calling DLBL from an assembler language program and using DSN and one or more data set names, you should supply an extended plist. For more information on how an assembler language program can supply an extended plist, see *z/VM: CMS Application Development Guide for Assembler*.

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. For more information on when the DLBL command is required, see *VSE/VSAM Programmer's Reference*.

2. The following special ddnames must be used to define VSE private libraries, and must be associated with the indicated logical units:

```
                Logical
     ddname      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
   ```

   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 for CMS/DOS Users: 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. For more information on when EXTENT information is required, see *VSE/VSAM Programmer's Reference*.

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 disk 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.

- 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

DMS331R  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. Because 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.

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.

MODE
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 data set ddname.

If no DLBL definitions with extent information are active, the following message is issued:

```
DMS324I  No user defined EXTENT in effect
```

Identifying Multivolume VSAM Extents for CMS/DOS Users: 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. For more information on when this type of EXTENT information is required, see *VSE/VSAM Programmer's Reference*.

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

DMS330R  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 25 disks may be specified; you do not need to specify them in alphabetic 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 data sets is provided:

DDNAME
The VSE file name or OS ddname.

MODE
The CMS disk file mode identifying one of the disks 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.

If no DLBL definitions with multiple volume specifications are active, the following message is issued:

```
DMS324I  No user defined MULT 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. Because a job catalog is in use, you must use the CAT option to indicate 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 6 on page 192.

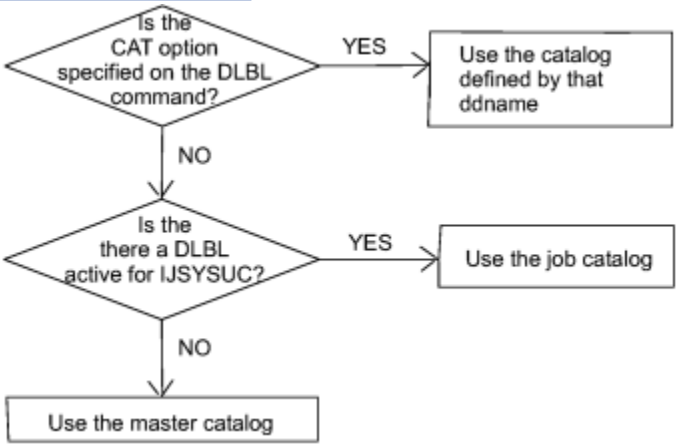


Figure 6. Determining Which VSAM Catalog to Use

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 DMS069E 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, see *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 for OS VSAM Users: 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. For more information on when EXTENT information is required, see *VSE/VSAM Programmer's Reference*.

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 disk 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.
- 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

  DMS331R  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. Because 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 for OS VSAM Users: 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. For more information on when this type of EXTENT information is required, see *VSE/VSAM Programmer's Reference*.

When you use the MULT option, you are prompted to enter additional disk file mode letters, as follows:

```
dlbl infile c (mult

DMS330R  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 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 25 disks may be specified; you do not need to specify them in alphabetic 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. Because a job catalog is in use, you must use the CAT option to indicate 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 6 on page 192.

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]]
```

```
DMS220R  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.

```
DMS320I  Maximum number of disk entries recorded
```

This message indicates 25 volumes have been specified for a VSAM data set, which is the maximum allowed under CMS.

```
DMS321I Maximum number of extents recorded
```

This message indicates 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.

```
DMS322I DDNAME ddname not found; no CLEAR executed
```

This message indicates the clear function was not performed because no DLBL definition is in effect for the ddname.

```
DMS323I  {Job|Master} catalog DLBL cleared
```

- This message indicates 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.

```
DMS330R  Enter volume specifications:
```

This message prompts you to enter volume specifications for existing multivolume VSAM files. For more information, see Identifying Multivolume VSAM Extents for CMS/DOS Users and Identifying Multivolume VSAM Extents for OS VSAM Users.

```
DMS331R  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. For more information, see Specifying VSAM Extent Information for CMS/DOS Users and Specifying VSAM Extent Information for OS VSAM Users.

Messages and Return Codes

- DMS001E No filename specified [RC=24]
- DMS003E Invalid option: *option* [RC=24]

- DMS005E No *option* specified [RC=24]
- DMS023E No filetype specified [RC=24]
- DMS048E Invalid filemode *mode* [RC=24]
- DMS050E Parameter missing after DDNAME [RC=24]
- DMS065E *option* option specified twice [RC=24]
- DMS066E *option1* and *option2* are conflicting options [RC=24]
- DMS069E Filemode *mode* not accessed [RC=36]
- DMS070E Invalid parameter *parameter* [RC=24]
- DMS086E Invalid DDNAME *ddname* [RC=24]
- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS221E Invalid data set name [RC=24]
- DMS301E SYS*aaa* not assigned for filemode *fm* [RC=36]
- DMS302E No SYSXXX operand entered [RC=24]
- DMS304E Invalid operand value *value* [RC=24]
- DMS305E Incomplete extent range [RC=24]
- DMS306E SYS*aaa* not assigned for IGNORE [RC=36]
- DMS307E Catalog DDNAME *ddname* not found [RC=24]
- DMS308E *mode* filemode in [non-]CMS format; invalid for [non-]CMS dataset [RC=24]

DOSGEN

Authorization

CMS Installer

Purpose

Use the DOSGEN command to load and save a saved segment (usually called CMSDOS) that contains the text files needed to create a CMS/DOS environment that simulates DOS/VSE (Disk Operating System/ Virtual Storage Extended) under CMS.

Operands

hexaddr
is the hexadecimal address where you want to load and save the saved segment. It must be greater than X'20000' and less than 16M. This address must match the address used on the CP DEFSEG command to define the saved segment.

segname
is the name you want to assign to the saved segment. It does not have to be CMSDOS, but it must match the name used on the CP DEFSEG command to define the saved segment.

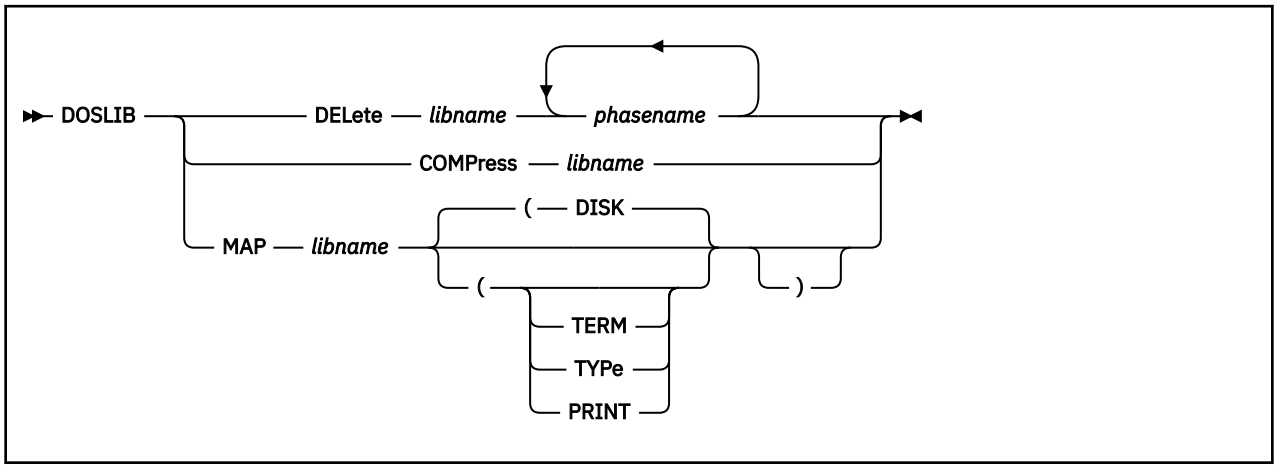
Usage Notes

1. Before the DOSGEN command can be used to load and save the saved segment, the CP DEFSEG command must be used to define the saved segment to CP.
2. To issue the DOSGEN command, you must be authorized to perform the CP SAVESEG operation.
3. DOSGEN performs the following operations:
   a. Checks the specified address contains valid characters and that it is greater than X'20000' and less than 16M
   b. Looks for a read/write accessed A-disk on which to write the CMS loader work file
   c. Loads all the text files needed for VSE simulation, starting at the address specified
   d. Assigns a storage protection key of X'D'
   e. Saves the saved segment
   f. Writes the load map to the A-disk as LOAD MAP A5.
4. Before you load and save CMSDOS, you must define your virtual machine with at least 512K free storage above the end of the CMSDOS saved segment. This provides room for the loader tables, which occupy the top pages of virtual storage. After you load and save the saved segment, the loader tables and the 512K free storage are no longer required.

Messages and Return Codes

- DMS006E No read/write A-disk accessed
- DMS095E Invalid address
- DMS111E DOSGEN failed due to LOAD errors
- DMS141S DOSGEN failed due to SAVESEG errors
- DMS412S DOSGEN failed due to SETKEY errors

DOSLIB



Authorization

General User

Purpose

Use the DOSLIB command to delete, compact, or list information about the executable phases in a CMS/DOS phase library.

Operands

DELete
deletes phases from a CMS/DOS phase library. The library is not erased when the last phase is deleted from the library.

COMPress
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.

phasename
is the name of one or more phases that exist in the CMS/DOS phase library.

Options

The following options specify the output device for the MAP function. If you specify more than one option, CMS uses the last option specified.

TERM
displays the MAP output at the terminal.

TYPe
displays the MAP output at the terminal. (This option has the same function as the TERM option).

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. The default is DISK.

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 you need.

Examples

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
  .         .       .
  .         .       .
  .         .       .
```

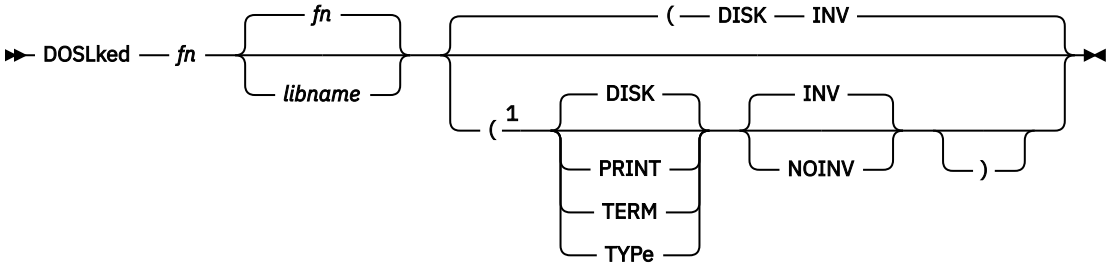
Messages and Return Codes

- DMS002E File *fn* DOSLIB not found [RC=28]
- DMS003E Invalid option: *option* [RC=24]
- DMS013W Phase *phase* not found in library *libname* [RC=4]
- DMS014E Invalid function *function* [RC=24]
- DMS037E Filemode *mode*[(*vdev*)] is accessed as read/only [RC=36]
- DMS046E No library name specified [RC=24]
- DMS047E No function specified [RC=24]
- DMS069E Filemode *mode* not accessed [RC=36]
- DMS070E Invalid parameter *parameter* [RC=24]
- DMS098E No phase name specified [RC=24]
- DMS104S Error *nn* reading file *fn ft fm* from disk or directory [RC=100]
- DMS105S Error *nn* writing file *fn ft fm* on disk or directory [RC=100]
- DMS213W Library *fn ft fm* not created [RC=4]
- DMS213W Library *libname* has no members [RC=4]
- DMS653E Error executing *command* rc=*nn* [RC=40]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in copying a file	“Messages and Return Codes” on page 90
Errors in erasing a file	“Messages and Return Codes” on page 217

DOSLKED



Notes:

[1] You can enter Options in any order between the parentheses.

Authorization

General User

Purpose

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).

Operands

fn
specifies the name of the source file or module to be link-edited. CMS searches for a:

- CMS file with a file type of DOSLNK
- Module in a private relocatable library (if IJSYSRL has been defined)
- CMS file with a file type of TEXT
- Module in the system relocatable library (if a mode was specified on the SET DOS ON command line)

The input *fn* is used as the default output *fn*.

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 which is the same as the input fn. The output file mode of the DOSLIB is determined if the:

- Libname DOSLIB exists on a read/write disk or directory, that file mode is used and the output is appended to it
- *fn* DOSLNK exists on a read/write disk or directory, *libname* DOSLIB is written to that disk or directory
- *fn* DOSLNK exists on a read-only extension of a read/write disk or directory *libname* DOSLIB is written to the parent

Note: If none of the above apply, *libname* DOSLIB is written to your disk or directory accessed as A

Options

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.

TYPe
displays the linkage editor map at your terminal. (This option has the same function as the TERM option.)

INV
indicates that if any DOSLKED control statements are found with a not valid operation, theses actions will be taken:

- VSE message 2101I and a copy of a not valid control statement will be displayed.
- VSE message 2101I and a copy of a not valid control statement will be included in the MAP file, unless NOMAP was specified in an earlier ACTION control statement.
- Processing will be terminated if CANCEL was specified on an earlier ACTION control statement.
- The DOSLKED command will issue a return code of 4 when it completes.

NOINV
means any DOSLKED control statements with a not valid operation are ignored.

Note: These control statements will be listed in the linkage editor map when ACTION MAP is in effect. This option allows update history information included within CMS text files to be reflected in the linkage editor map without producing an error. If no other errors are detected DOSLKED will complete with a return code of 0.

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 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. Before 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 before 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 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 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.

Examples

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.

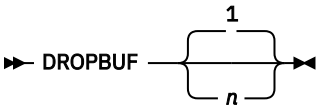
```
2101I INVALID OPERATION IN CONTROL STATEMENT
```

This message indicates a blank card was encountered in the process of link-editing a relocatable module. This message also appears in the MAP file. The not valid card is ignored and processing continues.

Messages and Return Codes

- DMS001E No filename specified [RC=24]
- DMS003E Invalid option: *option* [RC=24]
- DMS006E No read/write filemode accessed [RC=36]
- DMS007E File *fn ft fm* is not fixed, 80-character records [RC=32]
- DMS070E Invalid parameter *parameter* [RC=24]
- DMS099E CMS/DOS environment not active [RC=40]
- DMS104S Error *nn* reading file *fn ft fm* from disk or directory [RC=100]
- DMS105S Error *nn* writing file *fn ft fm* on disk or directory [RC=100]
- DMS210E Library *fn ft* is on a read/only filemode [RC=36]
- DMS245S Error *nnn* on printer [RC=100]
- DMS2521E DOSLKED cannot be performed on empty file *libname* [RC=88]

DROPBUF



Notes:

[1] The default is the last buffer created.

Authorization

General User

Purpose

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.

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 Notes

1. 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.

Messages and Return Codes

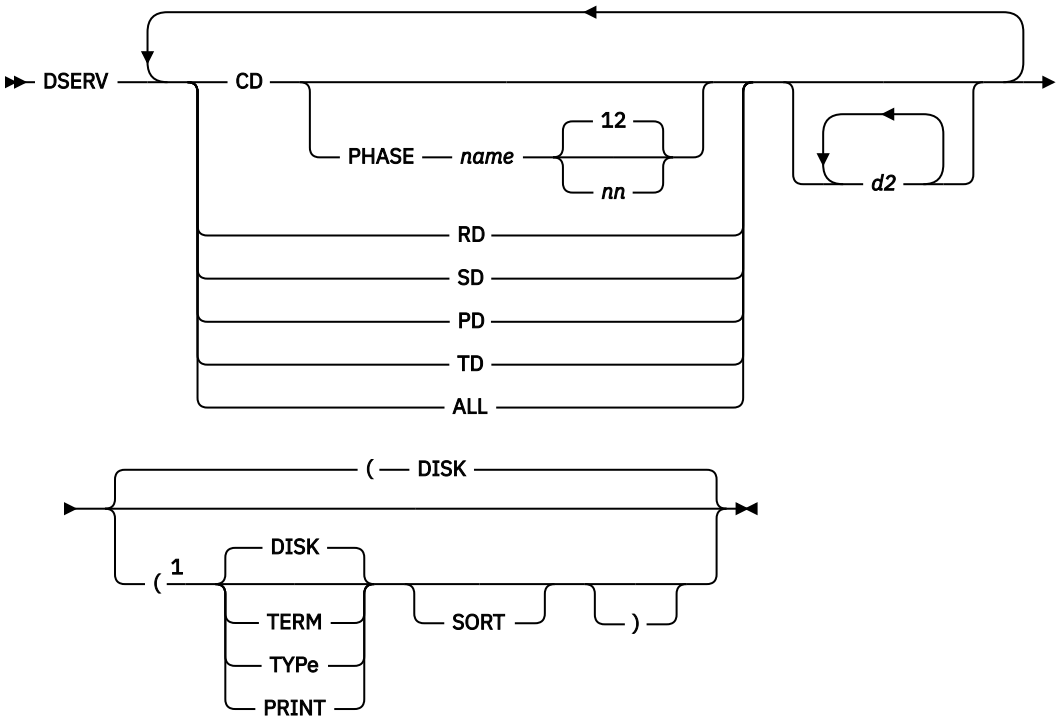
If an error occurs in DROPBUF processing, the return codes are:

RC
Meaning

1
Not valid number specified

2
Specified buffer does not exist

DSERV



Notes:

[1] You can enter Options in any order between the parentheses.

Authorization

General User

Purpose

Use the DSERV command in CMS/DOS to obtain information that is contained in VSE private or system libraries.

Operands

CD
RD
SD
PD
TD
ALL
specifies 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).

d2
indicates additional libraries whose directories are to be listed. For more information, see Usage Note “1” on page 207.

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.

TYPe
displays the output at your terminal. (This option has the same function as the TERM option.)

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 the 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 (by 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.

Examples

To display at your terminal an alphanumeric list of procedures cataloged on the system procedure library, you would issue:

```
dserv pd (sort term
```

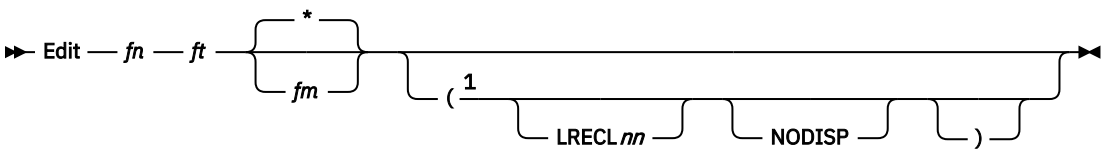
Responses

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

- DMS003E Invalid option: *option* [RC=24]
- DMS021W No transient directory [RC=4]
- DMS022W No core image directory [RC=4]
- DMS023W No relocatable directory [RC=4]
- DMS024W No procedure directory [RC=4]
- DMS025W No source statement directory [RC=4]
- DMS026W *phase* not in library [RC=4]
- DMS027E Invalid device *devtype* [for SYS*aaa*] [RC=24]
- DMS027W No private core image library [RC=4]
- DMS028W No {private|system} transient directory entries [RC=4]
- DMS047E No function specified [RC=24]
- DMS065E *option* option specified twice [RC=24]
- DMS066E *option1* and *option2* are conflicting options [RC=24]
- DMS070E Invalid parameter *parameter* [RC=24]
- DMS095E Invalid address *vstor* [RC=100]
- DMS099E CMS/DOS environment not active [RC=40]
- DMS105S Error *nn* writing file *fn ft fm* on disk or directory [RC=100]
- DMS245S Error *nnn* on printer [RC=100]
- DMS411S Input error code *nn* on SYS*aaa* [RC=*rc*]

EDIT



Notes:

[1] You can enter Options in any order between the parentheses.

Authorization

General User

Purpose

Use the EDIT command to invoke the z/VM XEDIT 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 more information on EDIT migration mode, see *z/VM: XEDIT Commands and Macros Reference*.

You can return control to the CMS environment by issuing the EDIT subcommands FILE or QUIT.

For more information on the EDIT subcommand formats and usage, use the online z/VM HELP Facility.

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 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 an accessed 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 disk or directory and its extensions are searched. If the file mode is not specified or is specified as an asterisk, all accessed disks or directories are searched for the specified file.

Creating new files

If the file mode is not specified, the editor assumed a file mode of A1.

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 these 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 not valid for the edit session.

Usage Notes

1. When you issue the EDIT command, XEDIT is called in EDIT migration mode.
2. The old CMS editor is no longer supported. Specifying (OLD on the EDIT command will cause an error message to be issued.

Examples

If you want to create a new file on your disk or directory accessed as A called OVERTIME DATA, 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:
```

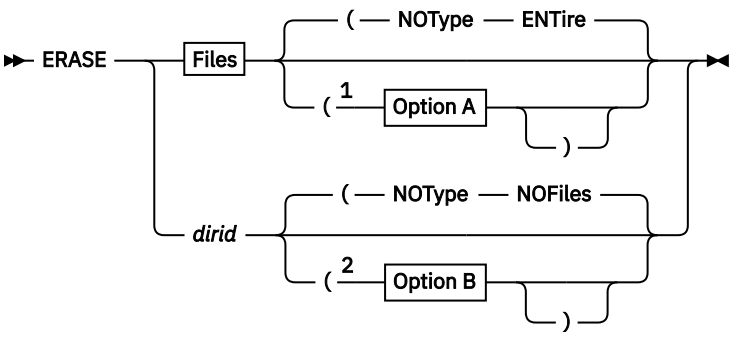
The input environment is entered by issuing the EDIT subcommands REPLACE or INPUT with no operands. All subsequent input lines are accepted as input to the file.

Messages and Return Codes

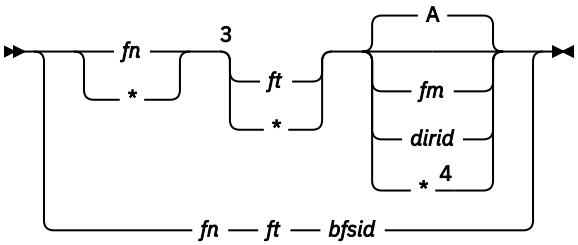
- DMS003E Invalid option *option* [RC=24]
- DMS024E File *fn ft fm* already exists [RC=28]
- DMS029E Invalid parameter *parameter* in the 'LRECL' option field [RC=24]
- DMS044E Record length exceeds allowable maximum [RC=32]
- DMS048E Invalid filemode *mode* [RC=24]
- DMS054E Incomplete fileid specified [RC=24]
- DMS069E Disk *mode* not accessed [RC=36]
- DMS076E Actual record length exceeds that specified [RC=40]

- DMS104S Error *nn* reading file *fn ft fm* from disk or directory [RC=100]
- DMS105S Error *nn* writing file *fn ft fm* on disk or directory [RC=100]
- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS117S Error writing to display terminal [RC=100]
- DMS132S File *fn ft fm* too large [RC=88]
- DMS143S Unable to load module [RC=40]
- DMS144S Requested file is in active status
- DMS151E 3278 MOD5 display terminal not supported by old CMS editor
- DMS2520E This level of CMS no longer supports the old CMS editor [RC=88]

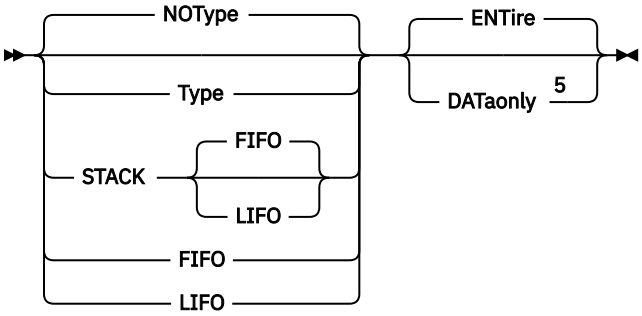
ERASE



Files



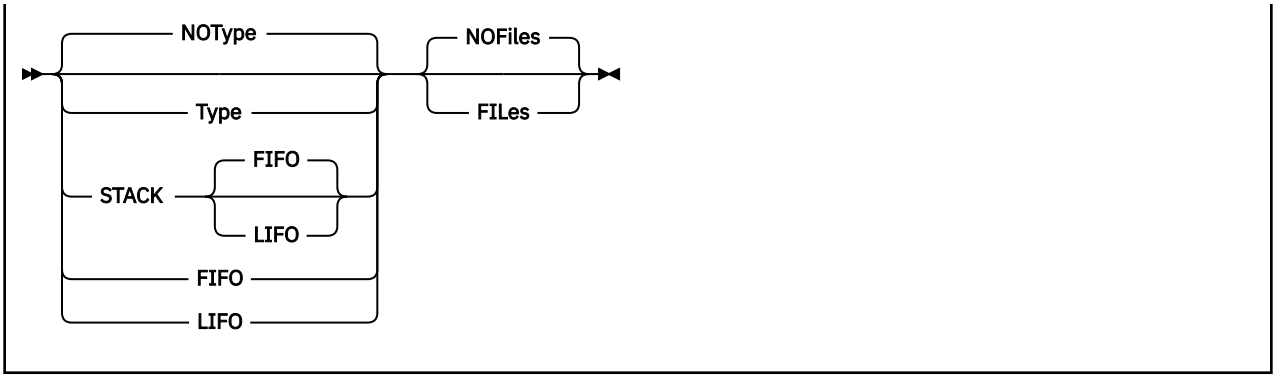
Option A



Notes:

[1] The defaults for Options A are shown above the line in the options group.
[2] You can enter Options in any order between the parentheses.
[3] If an asterisk is coded for both file name and file type, both a file mode letter and number are required.
[4] If an asterisk is coded for file mode, either the file name or file type must be specified by name.
[5] DATAONLY can only be used with SFS files.

Option B



Authorization

General User for SFS files and directories. A file pool administrator may enter this command for byte file system (BFS) files.

Purpose

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 or DIRWRITE authority
- One of your SFS directories.
- One of your byte file system (BFS) regular files

Operands

fn
*****
is the name of the files to be erased. An asterisk (*) coded in this position indicates all names are to be used.

ft
*****
is the file type of the files to be erased. An asterisk (*) coded in this position indicates all file types are to be used.

fm
*****
A
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 files with the specified name and file type are to be erased from all disks and directories accessed as read/write. If an asterisk is entered as the file mode, either the file name or the file type or both must be specified by name.

dirid
is either:

- the name of the directory containing the files to be erased, if you specify *fn ft* (fn and ft cannot both be specified as '*' on a single command invocation). 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.

For more information on *dirid*, see "Naming Shared File System (SFS) Directories" on page 6.

bfsid
identifies the byte file system (BFS).

For a BFS file, *fn* and *ft* are system-generated values that together uniquely identify the file within the BFS.

Options

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. The default is NOTYPE.

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

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

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

DATaonly
specifies the base file is not deleted from the directory, but all data in the file is erased. The existing file authorizations and aliases are retained. This results in an empty file with the specified name.

When this option is specified for an alias, all data in the base file will be deleted.

You cannot specify DATAONLY for:

- Directories
- External objects
- Erased or revoked aliases
- Minidisk files

ENTire
specifies the base file is deleted from the directory. All authorization granted to the file will also be lost. The default is ENTIRE.

When the specified file is an alias, it is deleted from the directory. The base file is not affected.

Note: When you issue ERASE for an external object, it and any control information stored in the Shared File System is erased, but the remote object itself is not affected.

FILes
indicates 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 valid only for erasing directories in your own directory structure.

NOFiles
indicates the directory must be empty of all files and subdirectories before it can be erased. It can contain erased and revoked aliases. The default is NOFILES.

Usage Notes

Applicable to both the shared file system (SFS) and byte file system (BFS)

1. You can erase a file or directory for which you have an UPDATE or EXCLUSIVE lock.
2. 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 (stops only messages 002 and 1184 from appearing)
- From an EXEC2 exec file and CMDCALL is not in effect
- From a REXX exec with ADDRESS COMMAND in effect

The TYPE option when specified will always display the erased file ID.

3. A successful ERASE command with the DATAONLY option will set the following file attributes to zero:
   - Number of data blocks
   - Number of records

Applicable to only the shared file system (SFS)

1. Table Table 12 on page 215 summarizes the interactions between the SFS object you are erasing, the authorities you must have to do so, and the results of the erase:

Table 12. Using the ERASE Command Options

Object being ERASED	Option "1.a" on page 215	Necessary authorities			Results
		File R	File W	Dir W	
File in file or directory control directory "1.b" on page 215	ENTIRE		●	●	File and all related authorizations and control data deleted.
File in file or directory control directory	DATAONLY		●		Contents of file deleted; aliases, authorizations, control data, and an empty file remain.
External object in file control directory	ENTIRE			●	External object deleted.
Alias (of base file in file or directory control directory) "1.c" on page 215	ENTIRE	● "1.f" on page 215		●	Alias deleted; base file unaffected.
Alias (of base file in file or directory control directory)	DATAONLY		● "1.d" on page 215		Contents of base file deleted.
Directory "1.e" on page 215	NOFILES	–	–	●	Directory is erased.
Directory "1.b" on page 215	FILES	–	●	●	Directory and all files it contains are erased.

Notes on the Table:

a. DATAONLY cannot be specified for subdirectories, external objects, erased or revoked aliases, or minidisk files.

b. You usually have the authority to erase files you own, unless SFS files are protected by an external security manager (ESM).

c. Includes erased and revoked aliases.

d. Authorizations refer to the base files.

e. You must either own the directory or have administration authority to erase it.

f. You must have file read authority to the base file to erase an alias in another user's directory.

2. 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.

3. Erasing several files at once:
   - 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
     ```

     If asterisks erase a set of files and an error occurs, the last nonzero return code is returned. If an irrecoverable 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.
   - 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.
   - If special characters are used to specify file name or file type, or both, processing continues for remaining file IDs that match the specified pattern.
4. ERASE inhibitors:

   You cannot erase a file in an SFS directory or erase an SFS directory if:
   - The file is open for writing by any user or open read-only by you. It can be open read-only by other users. An exception is that you can erase a file if you have it open by the FSOPEN, FSREAD, or FSWRITE macro.
   - The file or directory is locked by another user.
   - You or any user has a SHARE lock on the file or directory.

   You cannot erase a file or subdirectory of a directory control directory:
   - When you have the parent directory accessed read-only
   - Or when any other user has the directory accessed read/write.

   You cannot erase a directory control directory:
   - When you have the directory accessed read-only
   - Or when any other user has the directory accessed read/write.
5. 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 the owner has open.
6. You may also erase your open directory if you opened the directory with an intent of FILE.
7. To erase a directory that has files in it other than erased or revoked aliases (such as base files, aliases, or external objects), you must specify the FILES option.
8. 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.
9. To use the *fm* form of *dirid*, the directory must be accessed in read/write status. By default, other users' directories are accessed in read-only status. To access another user's directory in read/write status, specify the FORCERW option on the ACCESS command.
10. 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. To erase a file that is accessed in read/write status, enter:

    ```
    erase /
    ```

    If you are authorized to erase the file, but the directory is accessed in read-only status, you can still erase the file from the CMS command line or from the command column of the DIRLIST screen:
    - On the CMS command line, enter

      ```
      erase fn ft  dirid
      ```
    - On the DIRLIST screen, next to the name of the directory containing the file, enter

```
erase fn ft  /
```

This indicates you choose to erase the *fn ft* located in that directory.

By default, directories owned by other users are accessed in read-only status. To access another user's directory in read/write status, use the FORCERW option on the ACCESS command.

11. ERASE can be used on mixed case file IDs when FILELIST is entered with the MIXEDON option.

Applicable to only the byte file system (BFS)

1. Erasing a byte file system file with the CMS ERASE command erases the file and all hard links to that file.
2. Byte file system directories and special files may not be erased using the CMS ERASE command. Instead, use the OPENVM ERASE command.
3. ERASE may be used to delete a BFS regular file. However, the CMS short file name format of the file name must be used. You may not enter the ERASE command using the fully qualified byte file system (BFS) path name. Instead, use the OPENVM ERASE command.
4. You can only erase one byte file system file at a time with a single ERASE command; you cannot enter the asterisk (*) special character for the file name (*fn*) or the file type (*ft*).
5. A byte file system file cannot be erased if it is in use by an OPENVM user.

Examples

To erase all the files on your minidisk or directory accessed as A with a file type of SCRIPT, you would enter:

```
erase * script 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

- DMS002E File [*fn* [*ft* [*fm* | *dirname*]]] not found [RC=28]
- DMS037E Filemode *mode* is accessed as read/only [RC=36]
- DMS048E Invalid filemode *mode* [RC=24]
- DMS054E Incomplete fileid specified [RC=24]
- DMS069E Filemode *mode* not accessed [RC=36]
- DMS070E Invalid parameter *parameter* [RC=24]
- DMS071E Erase * * [fm|*] not allowed [RC=24]
- DMS109E Virtual storage capacity exceeded [RC=25]
- DMS389E Invalid *operandtype*: *operand* [RC=24]
- DMS1141W User filespace threshold still exceeded for file pool *filepoolid* [RC=0]
- DMS1161E Directory *dirname* contains subdirectories and thus cannot be erased. [RC=40]
- DMS1162E Directory *dirname* is not empty; specify FILES option [RC=40]
- DMS1163E The ERASE command failed for *fn ft fm* | *dirname* [RC=*nn*]

- DMS1180E You have an explicit lock on file *fn ft fm* | *dirname*; the erase failed [RC=70]
- DMS1180E You have an explicit lock on directory *dirname* or on an object in the directory; the erase failed [RC=70]
- DMS1181E Directory *dirname* contains an open file and thus cannot be erased [RC=70]
- DMS1184E File *fn ft fm* | *dirname* not found or you are not authorized for it [RC=28]
- DMS1184E Directory *dirname* not found or you are not authorized for it [RC=28]
- DMS1187E Too many subdirectory levels in *dirid* [RC=24]
- DMS1188E Filemode *fm* is not associated with a directory [RC=74]
- DMS1189E Filemode *fm* is associated with a top directory [RC=24]
- DMS1197E Directory *dirname* could not be opened; no files erased [RC=70]
- DMS1198E File *fn ft fm* | *dirname* is currently open; it must be closed before it can be erased [RC=70]
- DMS1198E Directory *dirname* is currently open; it must be closed before it can be erased [RC=70]
- DMS1199E You cannot erase a top directory [RC=88]
- DMS1223E There is no default file pool currently defined [RC=40]
- DMS1332E You are not authorized to erase one or more objects in the Directory *dirname* [RC=28]
- DMS2039E DATAONLY option is not supported on the current level of file pool *filepoolid* [RC=24]
- DMS2040E ERASE cannot be performed on a directory control directory that is accessed read-only [RC=36]
- DMS2500I File *fn ft fm* is already empty. [RC=0]
- DMS2510E Requested function is not supported for specified file object [RC=40]

Additional system messages may be issued by this command. The ERASE call type affects the message you may get as noted below. The reasons for these messages, the call type, and their location are:

Reason	Call type	Location
Errors in command syntax	X'0B' - X'0E'	“Command Syntax Error Messages” on page 1411
Errors in opening a file	X'0B' - X'0D'	“File Error Messages” on page 1418
Errors in the Shared File System	X'0B' - X'0E'	“File Pool Server Messages” on page 1414

ESERV

Authorization

General User

Purpose

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.

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 SYSPCH. If either SYSLST or SYSPCH is not assigned, the following files are created:

   Unit
   Output File

   SYSLST
   fn LISTING mode

   SYSPCH
   fn 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, if you assign:

   - SYSIPT to TAPE or READER, the source statements are read from that device.
   - 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. 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.

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

   ```
   exec eserv
   ```

8. 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 the ESERV program completed execution successfully. You may examine the SYSLST output to verify the results of the ESERV program execution.

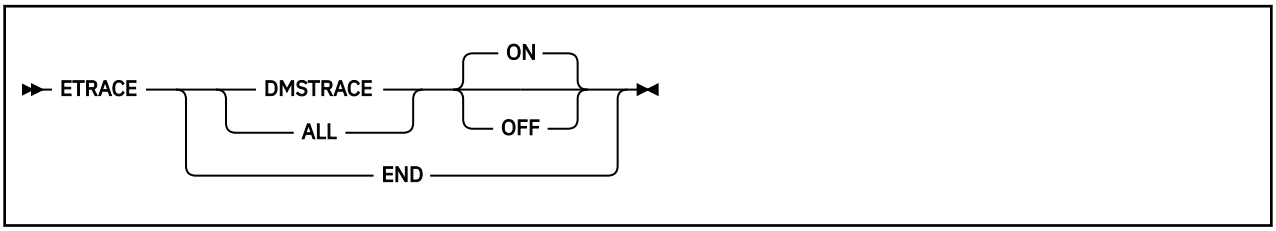
Messages and Return Codes

- DMS001E No filename specified. [RC=24]
- DMS002E File *fn ESERV* not found. [RC=28]
- DMS006E No read / write filemode accessed. [RC=36]
- DMS027E Invalid device*devtype* for *SYSaaa* [RC=24]
- DMS037E Filemode *mode* is read only. [RC=36]
- DMS070E Invalid argument *argument* [RC=24]
- DMS099E CMS/DOS environment not active. [RC=40]

Note: The ESERV EXEC calls other CMS commands to perform certain functions, so you may receive error messages that occur as a result of those commands.

For more information on the non-CMS error messages produced by the VSE ESERV program, see *Guide to the DOS/VSE Assembler*.

ETRACE



Authorization

General User

Purpose

Use the ETRACE command to enable/disable the recording of external events in a spool file in a virtual machine.

Operands

DMSTRACE
records the data passed by the DMSTRACE CSL routine, in a spool file.

ON
enables external tracing of events. This is the default option of the DMSTRACE operand.

OFF
disables external tracing of events and writes data in the buffer to the spool file.

ALL
enables/disables external tracing.

ON
enables external tracing. This is the default option of the ALL operand.

OFF
disables external tracing.

END
disables external tracing and writes the data to a spool file.

Usage Notes

1. The CP command CPTRAP or TRSOURCE must be issued first to make ETRACE effective. For more information, see *z/VM: CP Commands and Utilities Reference*.
2. Applications can record trace data through the DMSTRACE CSL routine only after ETRACE DMSTRACE has been enabled. For more information, see *z/VM: CMS Callable Services Reference*.

Messages and Return Codes

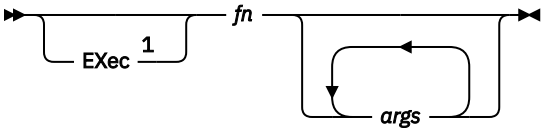
- DMS1330E CPTRAP/TRSOURCE must be enabled before calling ETRACE [RC=24]
- DMS1333E TRSOURCE is disabled. Buffer not written [RC=40]
- DMS1333E TRSOURCE is in EVENT mode. Buffer not written [RC=40]
- DMS1333S I/O or severe error. Buffer not written [RC=40]
- DMS1336E This function needs the CP Diagnose X'E0' command [RC=40]
- DMS1336E This function needs the CP Diagnose X'EC' command [RC=40]

- DMS1337E Insufficient storage to set up buffering [RC=104]
- DMS1338I ETRACE set [ON|OFF] for DMSTRACE [RC=0]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

EXEC



Notes:

[1] Keyword EXEC may be omitted as long as SET IMPEX is set on.

Authorization

General User

Purpose

Use the EXEC command to execute one or more CMS commands or exec control statements contained in a specified REXX, CMS EXEC, or EXEC 2 file.

Operands

EXec
this keyword may be omitted if you are executing the exec procedure from the CMS command environment and have not issued the command SET IMPEX OFF. (SET IMPEX ON, the default, means CMS treats exec files as commands, making it unnecessary for you to use the EXEC command to execute them.)

fn
is the file name of a file containing one or more CMS commands 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 a CMS editor creates have, by default, variable-length, 80-character records.

args
are any arguments you choose 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 REXX 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 REXX are handled differently from the way they are in EXEC or EXEC 2. For more information, see *z/VM: REXX/VM Reference*.

For information on REXX, see *z/VM: REXX/VM Reference* and *z/VM: REXX/VM User's Guide*.

For information about EXEC 2 control statements, use the HELP facility to get complete descriptions of EXEC and EXEC 2 control statements, special variables, and built-in functions by entering:

```
help exec menu
```

or

```
help exec2 menu
```

Examples

If the implied EXEC function is set to OFF (QUERY IMPEX to find out the setting), and if you want to execute your SQUARE EXEC (which squares a number), enter:

```
exec square 4
```

If IMPEX is on, you can simply enter:

```
square 4
```

For more examples on using the EXEC command, see *z/VM: CMS User's Guide*.

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
specifies the return code from the CMS command.

The amount of information displayed during the execution of:

- A REXX file depends on whether tracing is set on. For more information, see *z/VM: REXX/VM Reference*.
- An EXEC 2 file depends on the setting of the &TRACE control statement. For more information about &TRACE is available through the z/VM Help Facility by entering:

  ```
  help exec2 &trace
  ```

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 HCPLNK053E *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

- DMS175E Invalid EXEC command RC=10000
- DMS255T Insufficient storage for Exec interpreter RC=10096
- DMS240E Alternate exec processor *name* not found RC= -3
- DMS240E Unable to load the CMS exec processor "DMSEXT" RC= -3
- DMS639E Error in *routine-name* routine; return code was *nn*
- DMS1229E *fileid* is empty [RC=88]
- DMS042E No execid specified [RC=24]

If the CMS EXEC interpreter finds an error, it displays the message:

```
DMS072E  Error in EXEC file fn, line nnn:
message
```

The possible errors, and the associated return codes, are:

Description	Return Code
FILE NOT FOUND	801

Description	Return Code
&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:

```
DMS085E  Error in fn ft fm, line nnn - message
```

The possible errors and the associated return codes related to this message 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

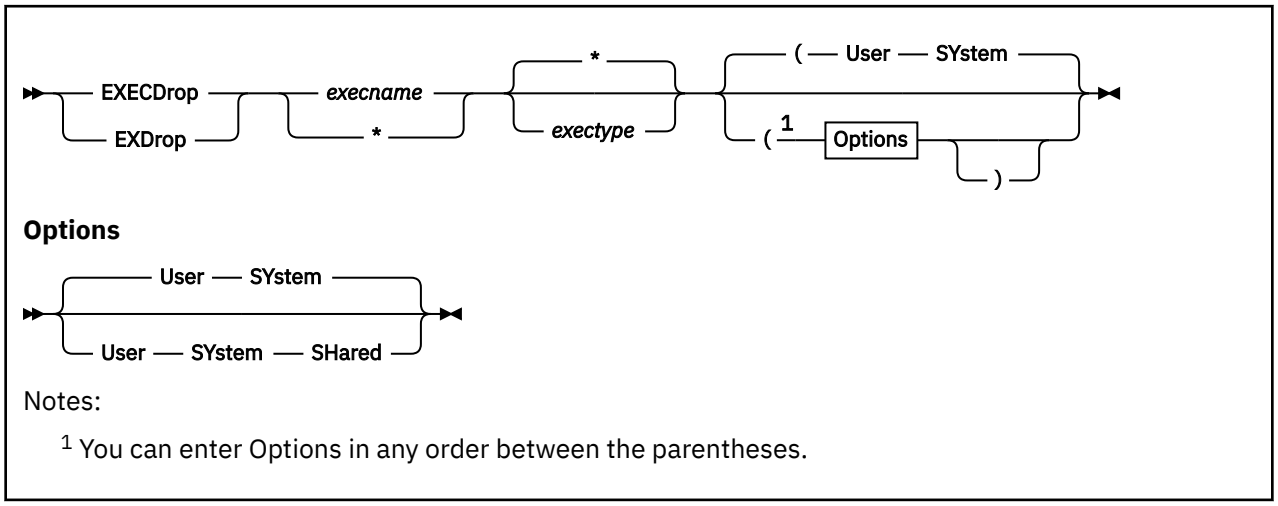
Description	Return Code
END OF FILE FOUND IN LOOP	10015
DIVISION BY ZERO	10016
INVALID LOOP CONDITION	10017
ERROR RETURN DURING &ERROR ACTION	10019
ASSIGNMENT TO UNSET ARGUMENT	10020
STATEMENT OUT OF CONTEXT	10021
INSUFFICIENT STORAGE AVAILABLE	10097
FILE READ ERROR nnn	10098
TRACE ERROR nnn	10099

For more information on the possible errors and associated return codes for REXX, see *z/VM: REXX/VM Reference*.

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in the Shared File System	“File Pool Server Messages” on page 1414

EXECDROP



Authorization

General User

Purpose

Use the EXECDROP command to remove the specified exec(s) or XEDIT macro(s) from storage or to discontinue use of the specified exec(s) or editor macro(s) in a CMS installation saved segment.

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 (*).

Options

User
indicates the storage for the exec(s) was allocated from user free storage. Only the execs that satisfy the execname, the exectype, or both qualifications and that also have the USER attribute are dropped. If neither USER, SYSTEM, nor SHARED is specified, all execs with the USER or SYSTEM attributes that satisfy the execname, the exectype, or both qualifications are dropped. This is the default.

SYstem
indicates the storage for the exec(s) was allocated from nucleus free storage. Only the execs that satisfy the execname, the exectype, or both qualifications and that also have the SYSTEM attribute are dropped. If neither USER, SYSTEM, nor SHARED is specified, all execs with the USER or SYSTEM attributes that satisfy the execname, the exectype, or both qualifications are dropped. This is the default.

SHared
indicates the exec(s) was located in a CMS installation saved segment.

Only the execs that satisfy the execname, the exectype, or both qualifications and also have the SHARED attribute are dropped for the duration of your CMS session.

If neither USER, SYSTEM, nor SHARED is specified, all execs with the USER or SYSTEM attribute that satisfy the execname, the exectype, or both qualifications are dropped.

SHARED execs can only be dropped when SET INSTSEG is ON. To discontinue use of all SHARED execs temporarily, use the SET INSTSEG OFF command.

Usage Notes

In XA and XC virtual machines, the EXECDROP command operates on execs and macros loaded above the 16MB line with the EXECLOAD command.

Examples

To drop all storage-resident execs which were loaded with the SYSTEM or USER attribute, 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

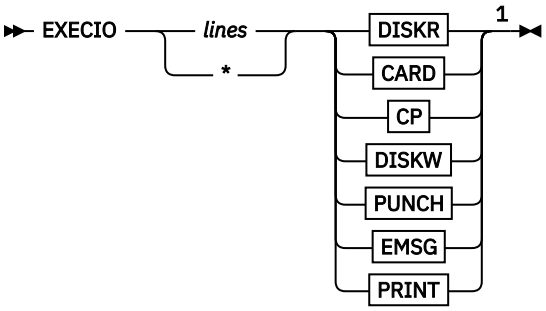
- DMS070E Invalid parameter *parameter* [RC=24]
- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS416W There are no [*execname exectype*] {system|[or]user|[or]shared} EXECs storage resident [RC=28]
- DMS418W Drop pending for *execname exectype* [RC=4]

The reasons for these messages and their location are:

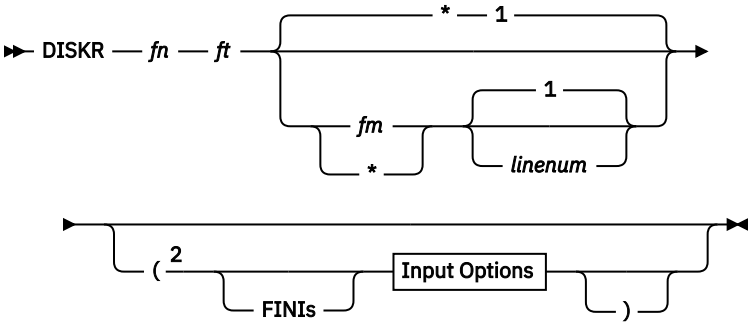
Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

EXECIO

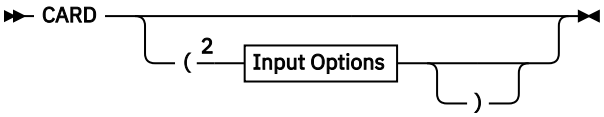
Note: You may find existing code in the z/VM product or developed in your installation that uses EXECIO functions. CMS Pipelines offers the same functionality in a more flexible and efficient way. Refer to "Appendix C" of the *z/VM: CMS Pipelines User's Guide and Reference*, SC24-6252 for a brief overview of CMS Pipelines device drivers to use instead.



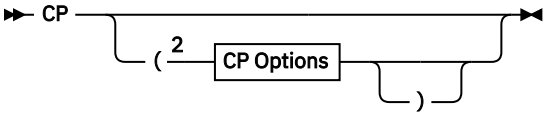
DISKR



CARD



CP

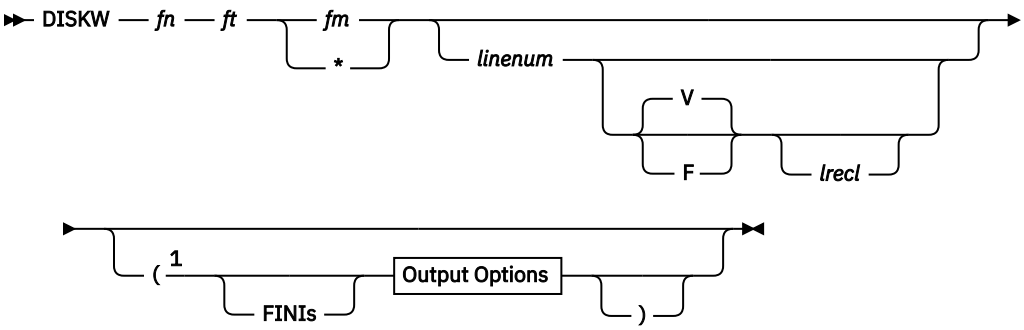


Notes:

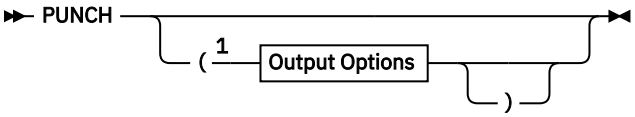
[1] If a right parenthesis marks the end of an EXECIO option, it must be preceded by at least one blank character. A right parenthesis cannot be used to mark the end of the STRING option.

[2] You can enter Options in any order between the parentheses.

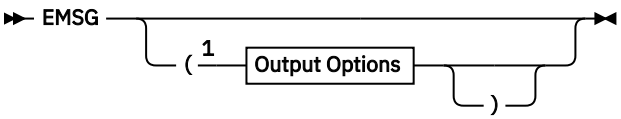
DISKW



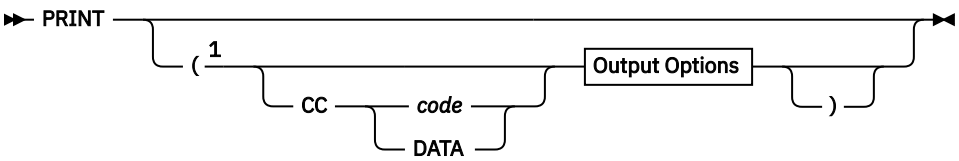
PUNCH



EMSG



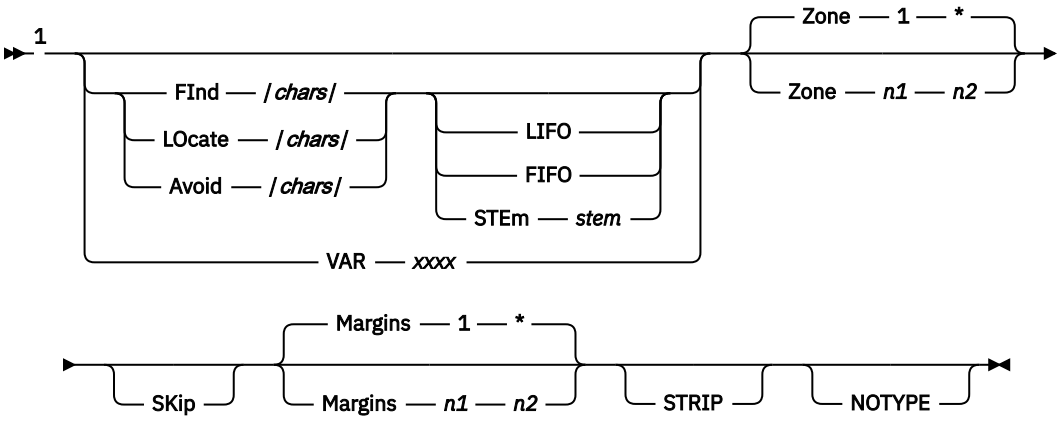
PRINT



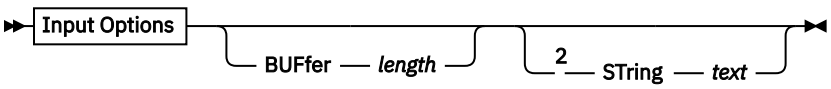
Notes:

[1] You can enter Options in any order between the parentheses.

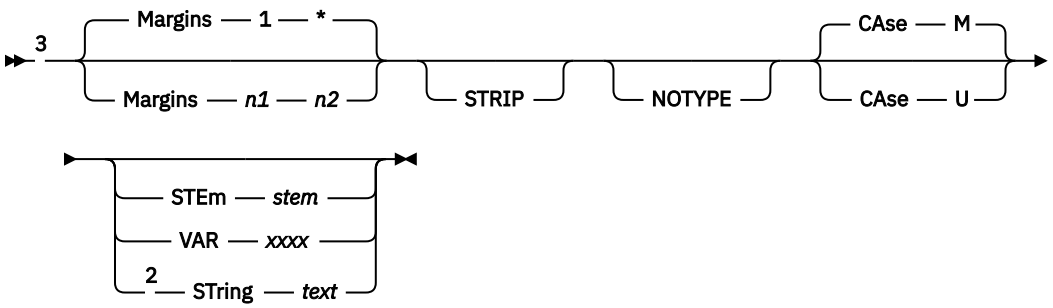
Input Options



CP Options



Output Options



Notes:

[1] Input options control how data is passed into your exec from EXECIO.
[2] If the STRING option is used, it must be the last option specified.
[3] Output options control how data is passed from your exec to EXECIO.

Authorization

General User

Purpose

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. You should be familiar with use of execs under REXX or EXEC 2, to gain the full benefit from the EXECIO command. For more information about REXX, see *z/VM: REXX/VM Reference*.

Operands

lines
is the number of source lines processed. This can be any nonnegative integer. With the VAR option, the number of lines must be 1. An asterisk (*) indicates the operation is to terminate when:

- A null (0-length) line is read during an *output* operation.
- 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.

Note:

relative line number
specifies the number of lines processed to satisfy an EXECIO operation.

absolute line number
specifies the number of the line relative to the top of the file.

DISKR
reads 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 REXX variable if the STEM or VAR options are specified.

fn
specifies the file name of the file.

ft
specifies the file type of the file.

fm
specifies the file mode of the file. When file mode is specified, that disk or directory and its extensions are searched. If file mode is optional and is not specified, or is specified as an asterisk (*), all accessed disks and directories are searched for the specified file. If file mode is required and an asterisk (*) is specified, the file with the specified file name and file type on the first accessed mode (in alphabetic order) will be opened. If no file is found that matches, EXECIO will fail.

Because CMS checks for open files first, if you specify an asterisk for *fm*, you may get unexpected results if there are open files matching the file name and file type specified.

linenum
is the absolute line number within the specified file where a DISKR or DISKW operation is to begin. If *linenum* has a value of zero, or is 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. If a value is zero or is not specified for newly opened files, reading begins at the first line. 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, a *linenum* value must be specified explicitly.

CARD
reads a specified number of lines from the virtual reader to the program stack (FIFO) or to an EXEC 2 or REXX 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 REXX 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 by means of the:

- STRING option on the EXECIO command line
- Next line from the program stack

Remember, all characters of CP commands must be in upper case.

DISKW
writes a specified number of lines from the program stack or from an EXEC 2 or REXX 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. For more information, see DISKW Operand.

fn
is as described under the DISKR operand.

ft
is as described under the DISKR operand.

fm
is as described under the DISKR operand.

linenum
is as described under the DISKR operand.

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 bypasses the use of the program stack, in which case the maximum is the smaller of the limit determined by the amount of virtual storage available in your machine, or the limit defined by the CMS file system (2^{31}-1). If recfm or lrecl is specified for the DISKW operation, a *linenum* value must be specified explicitly.

PUNCH
transfers a specified number of lines from the program stack or from an EXEC 2 or REXX variable if the STEM or VAR options are specified to the virtual punch.

EMSG
causes a message to be displayed. The text of the message may be the:

- Character string specified on the STRING option
- Next available line(s) from the program stack
- Information from a STEM or VAR variable

Messages are edited according to the current CP SET EMSG settings. However, the severity code of the message is ignored when editing the message.

PRINT
transfers a specified number of lines from the program stack or from an EXEC 2 or REXX 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 the carriage control character is the first byte of each line. You cannot specify a blank for the carriage control character, but if you omit the CC operand, a blank code is used as the default carriage control character. If CC is the last option specified and it is not followed by a code or DATA, the default is DATA.

Note: For more information about carriage control and printing from the program stack, see CC Operand.

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 the first byte of each line sent to the virtual printer is a carriage control character.

Options

Avoid
is like the LOCATE option, except the search is for a line (or zone portion of that line) that *does not* contain the specified characters.

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 though 2^{31}-1. If this option is not specified, up to 8192 characters of the response are returned.

CAse
causes data read from the program stack, from a STEM, or from a variable to be:

- Translated to uppercase if U is specified
- Not translated (mixed) if M is specified

The default value is M (mixed).

FInd
writes the following, LIFO (last-in first-out) to the program stack or FIFO (first-in first-out) to an EXEC 2 or REXX array:

1. The contents of the line that *begins with* the characters (*chars*) specified between delimiters
2. The line number of the first occurrence of that line (or zone portion of that line). For DISKR operations, both the relative and absolute line numbers are written. Otherwise, only the relative line number is written. FIND is case-sensitive.

If you choose to *search* only a portion of each line, use the ZONE option, explained below. If you choose 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.

Note: You must specify a space between the FIND operand and the first delimiter.

FINIs
causes the specified file to be closed following completion of a DISKR or DISKW operation.

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).

LOcate
is like the FIND option, except the object characters can occur any place within a line (or zone portion of that line), as opposed to only at the beginning of the line, as with the FIND option.

Margins
specifies 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.

NOTYPE
suppresses the display of message DMSEIO632E at the virtual console.

SKip
allows a read function (DISKR, CARD, CP) to occur without writing any information to the program stack.

STEm *stem*
indicates the specified *stem* defines variables used either to supply input data for EXECIO output-type operations (PUNCH, PRINT, EMSG, and DISKW), or as the destination for output for EXECIO input-type operations (CARD, DISKR, and CP). The variables defined will consist of the stem you specify concatenated with a number. For example, if the *stem* is *xxxx*, the variable *xxxx0* will contain the number of lines returned, *xxxx1* will contain the first line returned, *xxxx2* will contain the second and so on.

Note: The *stem* can be a regular REXX stem of the form *xxx.*. For more information on the REXX stems, see *z/VM: REXX/VM Reference*. For example, if the REXX stem is *xxx.*, the variable *xxx.0* will represent the number of lines returned for input-type operations; *xxx.1* will contain the first line returned, *xxx.2* will contain the second, and so on. REXX stems and variables must be uppercase.

If the STEM option is used on a print operation, channel 9 or channel 12 indications returned from the hardware will be ignored.

The maximum length variable name with the STEM option is 240 bytes.

STring
supplies up to 255 characters of 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.

STRIP
specifies trailing blank characters are to be removed from any output lines or lines returned.

VAR *xxxx*
indicates the *xxxx* supplies the input data for output-type operations (PUNCH, PRINT, EMSG, and DISKW) or that *xxxx* is the destination for output for the input-type operations (CARD, DISKR, and CP). If VAR is specified, the number of lines must be 1. The maximum length variable name is 250 characters. The variable *xxxx* must be uppercase.

If EXECIO returns a nonzero return code greater than 3, any variables specified by STEM or VAR will be undefined and cannot be used.

Zone
restricts 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.

If EXECIO returns a nonzero return code greater than 3, any variables specified by STEM or VAR will be undefined and cannot be used.

Usage Notes

Extended Descriptions and Use Information

1. EXECIO commands are usually issued as statements from REXX 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 REXX and EXEC 2 supply a tokenized parameter list and, if necessary, an extended parameter list. When EXECIO is passed an extended parameter list, numbers in that list are truncated on the right after 10 digits.
2. Remember 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.
3. For write operations, data to be written is usually taken from the program stack. However, data to be written may be supplied by the STRING option or by the VAR or STEM options (in all cases the exec in question must be a REXX exec or an EXEC 2 exec).
4. If the STEM option is used for a print operation, any channel 9 or channel 12 indications returned by the hardware will be ignored. When channel 9 or channel 12 is detected by the hardware, it will inform the issuer by means of a return code. A return code of 102 indicates channel 12 was sensed; a return code of 103 indicates channel 9 was sensed.
5. If the issuer of the EXECIO print operation needs to be able to handle channel 9 or 12 indications, the issuer should use the VAR, STACK, or STRING operand rather than STEM.

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. Usually 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, and so forth) 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, the QUEUED() built-in function can be used to return the same information as the CMS SENTRIES command. For more information on the QUEUED() built-in function, see *z/VM: REXX/VM Reference*.

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, the stacked lines may remain in the program stack and you may have to take extra actions. For more information, see CC Operand.

Setting Variables Directly

The VAR and STEM options allow EXEC 2 and REXX 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, and so forth) 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, the *xxxx0* contains the number of lines of data returned.

For VAR and STEM operations called from REXX, the maximum length of the value that EXECIO processes is 2^{24}; if called from an EXEC 2 program, the maximum length of the value is 255. If the command returns a nonzero 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. If you choose, you may close these files manually by invoking the CMS FINIS command. For files in an SFS directory, changes to all files and directories open on the same work unit are committed, and the files or directories remain open. This is a coordinated commit, meaning all changes to protected resources on the work unit are committed in unison (or rolled back, if any of the resources cannot commit). Using the FINIS option (or the FINIS command) to close any open files will commit your changes. However, you must check the return code to verify updates have been made, particularly when your application is using SFS files. A nonzero return code could mean the work unit has been rolled back and the changes have been lost.

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 00E
```

or

```
CP SPOOL PRINTER 00E CLOSE
```

can be used to close the virtual printer after using the EXECIO PRINT function.

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. For more information on the CP CLOSE command, see *z/VM: CP Commands and Utilities Reference*.

If you have specified the PRINT or PUNCH operand and you try to write a line 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 files are processed, see the CP CLOSE and CP SPOOL commands in *z/VM: CP Commands and Utilities Reference*.

DISKR Operation

The first line read on a DISKR operation may be:

- The first line of the specified file
- Specified using the *linenum* operand
- Determined by the results of a previous operation

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 on a line basis and an error code is returned. If you specify the BUFFER option, data is truncated on a line basis after the number of characters specified in *length* or 8192 is reached, whichever is greater. Each line returned ends with a X'15' character. This must be allowed for when calculating the buffer size needed. 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 by 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, 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 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 see the CP SET command in *z/VM: CP Commands and Utilities Reference*.

Remember 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.

Note: If asynchronous commands are issued the response is returned to the console, not the buffer.

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, for example, default *linenum* when writing to a newly opened file
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
```

Note: When using the *linenum* operand to insert lines in the middle of CMS variable length files, be aware of the way CMS handles these files. Any variable length line inserted **must** be equal in length to the line it displaces. Otherwise, all lines following the one inserted are truncated.

For example, if the variable format file WORDS LEARNING A is:

```
A is for apple
C is for cake
C is for candy
D is for dog
K is for Kate
M is for Mary
S is for Sarah
```

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 and 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 you may specify is 255 unless the VAR or

STEM option bypasses the use of the program stack, in which case the maximum is the smaller of the limit determined by the amount of virtual storage available in your machine, or the limit defined by the CMS file system (2^{31}-1).

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.

When creating a new file, the lrecl operand is only processed when the EXECIO command writes to the file. If the lines operand is 0, the lrecl operand is ignored. The first EXECIO command that writes to the new file (lines operand is greater than 0) will set the lrecl.

For example, if an exec is coded as

```
EXECIO 0 DISKW NEW FILE A 0 F 100
EXECIO 1 DISKW NEW FILE A (STRING line of the file
```

NEW FILE will be created as recfm=F and lrecl=80. The lrecl of 100 from the first EXECIO command is ignored because the lines operand is 0. The second EXECIO command sets the lrecl and 80 is chosen because it is the default lrecl for recfm=F files.

If an exec is coded as

```
EXECIO 1 DISKW NEW FILE A 0 F 100 (STRING line of the file
```

then NEW FILE will be created as recfm=F and lrecl=100.

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. For more information on the PRINTL macro, see *z/VM: CMS Macros and Functions Reference*.

If you are using EXECIO with either the VAR, STACK, or STRING operand, and the virtual FCB defines channel 9 or channel 12, it may be necessary to reset the carriage control. When channel 9 (return code 103) or channel 12 (return code 102) is sensed, the write operation terminates after carriage spacing, but before writing the line. If you are printing from the program stack, the line will remain in the stack. The carriage control character should be modified to take the appropriate action (for example, skip to channel, or print with no space).

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 this format:

```
xxxmmmnnns
```

Where:

xxxmmm
specifies the issuing module name

nnn
specifies the message number

s
specifies the message type:

E
Error

I
Informational

W
Warning

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:

- Four lines are to be read from MYFILE DATA, starting at line 1.
- 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
            .
```

Two situations that would cause the second EXECIO command to not begin execution at line 5 are:

- A program other than EXECIO accessing MYFILE DATA after the first and before the second EXECIO command is executed.
- 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 (as would issuing the CMS FINIS command on MYFILE DATA). 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 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
```

Remember that accessing MYFILE DATA by another program or causing a CMS READY message to be displayed before 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 by means of 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

The STEM option lets an array of EXEC 2 or REXX variables be set directly, bypassing the stack. For example, if you want the first 3 lines of MYFILE DATA to be assigned to REXX 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 REXX variables, the variable name should be enclosed in quotation marks and must be in uppercase.

If you want the first 3 lines of MYFILE DATA to be assigned to EXEC 2 variables Y1, Y2, and Y3, issue:

```
'EXECIO 3 DISKR MYFILE DATA * 1 (STEM Y'
```

Variable Y1 now contains the first line of MYFILE DATA, variable Y2 contains the second line, and variable Y3 contains the third line. Variable Y0 contains the number of lines (in this example, 3). For EXEC 2 variables, you can omit the ‘&’ from the variable name, quotation marks are not necessary, and you can use mixed case.

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 REXX 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 REXX variables, the variable name should be enclosed in quotation marks and must be in uppercase. For EXEC 2 variables, you omit the ‘&’, quotation marks 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. For information on using EXECIO within REXX programs, see *z/VM: CMS User's Guide*.

Messages and Return Codes

- DMS044E Record length exceeds allowable maximum [RC=132]
- DMS257T Internal system error at address *addr* (offset *offset*)
- DMS621E Bad Plist: Device and lines arguments are required [RC=24]
- DMS621E Bad Plist: Invalid value *value* for number of lines [RC=24]
- DMS621E Bad Plist: Missing DEVICE argument [RC=24]
- DMS621E Bad Plist: Invalid DEVICE argument *argument* [RC=24]
- DMS621E Bad Plist: Disk *argument* argument is missing [RC=24]
- DMS621E Bad Plist: Invalid character in file identifier [RC=24]
- DMS621E Bad Plist: Invalid value *value* for disk file line number [RC=24]
- DMS621E Bad Plist: Disk filemode required for DISKW [RC=24]
- DMS621E Bad Plist: Invalid record format *recfm*-- Must be either F or V [RC=24]
- DMS621E Bad Plist: Invalid record length argument *lrecl* [RC=24]
- DMS621E Bad Plist: File format specified *recfm* does not agree with existing file format *recfm* [RC=24]
- DMS621E Bad Plist: File lrecl specified *lrecl* does not agree with existing file lrecl *lrecl* [RC=24]
- DMS621E Bad Plist: Invalid positional argument *argument* [RC=24]
- DMS621E Bad Plist: EXECIO options only allowed with extended plist [RC=24]
- DMS621E Bad Plist: Unknown option name *name* [RC=24]
- DMS621E Bad Plist: Value missing after *option* option [RC=24]
- DMS621E Bad Plist: Value *value* not valid for *option* option [RC=24]
- DMS621E Bad Plist: *option* option is not valid with *option* option [RC=24]
- DMS621E Bad Plist: *option* option not valid with *operation* operation [RC=24]
- DMS621E Bad Plist: STRING option with LINES=* is valid only for CP operation [RC=24]
- DMS621E Bad Plist: VAR option with LINES>1 is invalid [RC=24]
- DMS621E Bad Plist: Invalid mode *mode* [RC=24]
- DMS621E Bad Plist: Invalid EXEC variable name [RC=24]
- DMS621E option *option* can only be executed from an EXEC 2 or REXX exec [RC=4]
- DMS621W Bad Plist: option *option* is ignored with operation *operation* [RC=0]
- DMS621W Bad Plist: option *option* is ignored with option *option* [RC=0]
- DMS621E Bad Plist: Option *option* can only be executed from an EXEC 2 or REXX exec [RC=*24*]
- DMS622E Insufficient free storage for EXECIO [RC=*rc*]
- DMS632E I/O error in EXECIO: rc=*nnnn* from *command* command [RC=*rc*]
- DMS1144E Implicit rollback occurred for work unit *workunitid* [RC=31]
- DMS1252T Rollback unsuccessful for file pool *filepoolid*

Additional system messages may be issued by this command. The reasons for these messages and their location are:

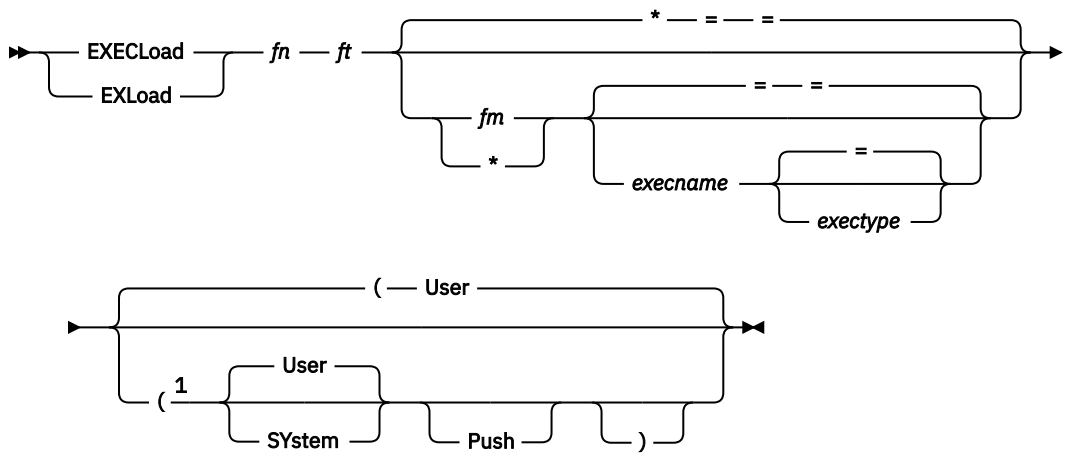
Reason	Location
Errors in the Shared File System	“File Pool Server Messages” on page 1414
Errors in using a file	“File Error Messages” on page 1418

Return codes:

RC	Meaning
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 file pool repository
1*nn*	100 + return code from I/O operation (if nonzero)
2008	Variable name supplied on STEM or VAR option was not valid
nnnn	2000 + return code from EXECCOMM command (if nonzero)
*x*1*nnn*	1000 + return code from CP command (if nonzero) Where: *x* is 0, 1, 2, or 3
1*xnnnn*	100000 + return code from CP command (if nonzero) Where: *x* is 0, 1, 2, or 3

Note: For more information on the EXECCOMM function and the associated return codes, see *z/VM: REXX/VM Reference*.

EXECLOAD



Notes:

[1] You can enter Options in any order between the parentheses.

Authorization

General User

Purpose

Use the EXECLOAD command to load an exec or XEDIT macro into storage and prepare it for execution.

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 (*). You must specify *fm* (or *) if you want to specify an execname and exectype.

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 the storage for the loaded exec is allocated from user free storage. This is the default.

SYstem
specifies the storage for the loaded exec is allocated from nucleus free storage.

Push
specifies 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.

Usage Notes

1. The file name and file type can each be 1-8 characters. The valid characters are A-Z, a-z, 0-9, $, #, @, +, - (hyphen), : (colon), and _ (underscore). The execname and exectype of the execid can also be 1-8 characters. However, the execname and exectype are not limited to the file name and file type character set. The only characters that are not valid 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. The amount of storage required to load an exec includes system overhead for the control blocks and I/O buffers required to maintain the exec in storage.
5. In XA and XC virtual machines, you can load a REXX exec or macro above the 16MB line.
6. When an EXEC or XEDIT macro has been EXECLOADed into storage and the EXEC or XEDIT macro is invoked through a CMSCALL on which a FBLOCK is supplied, the high–order bit of the FBLOCK address must be set on in order for the usage count reported by the EXECMAP command to be incremented. For more information on File Block, see *z/VM: REXX/VM Reference*.

Examples

The following command:

```
execload tphone exec a (user
```

loads the TPHONE EXEC from your disk 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

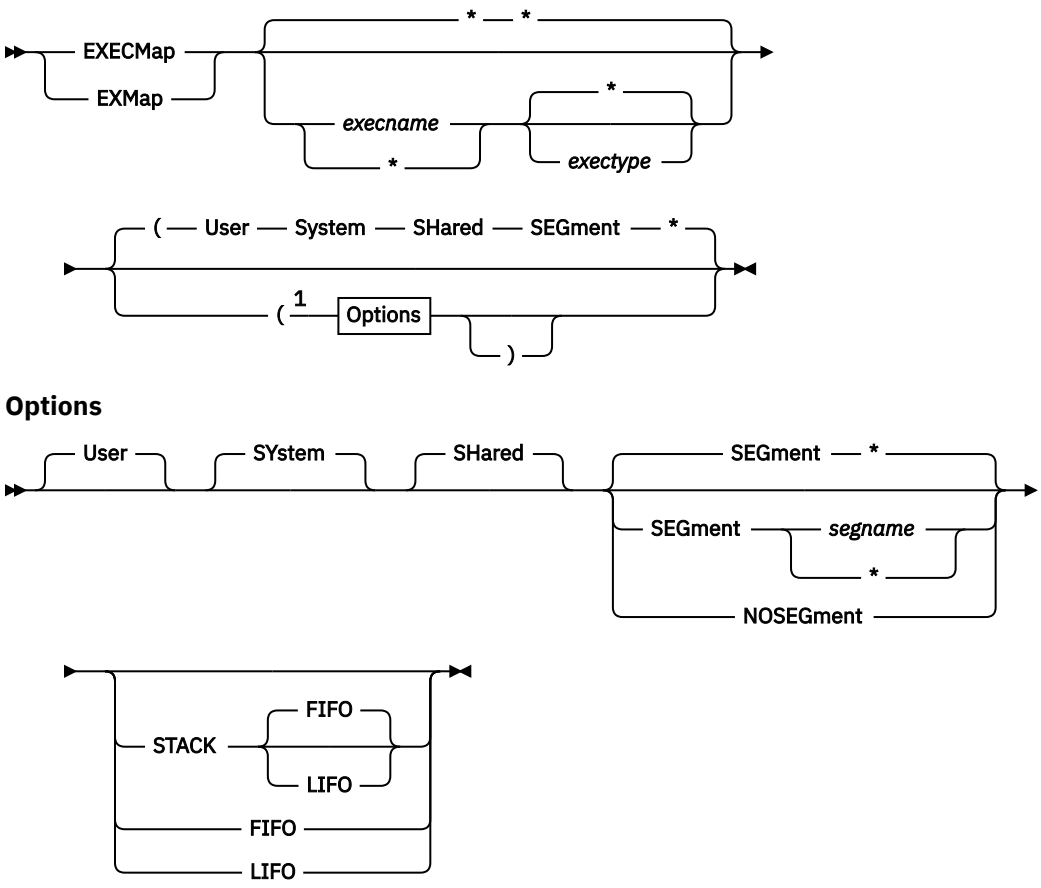
- DMS030E File *fn ft fm* is already active [RC=37]
- DMS070E Invalid parameter *parameter* [RC=24]
- DMS104S Error *nn* reading file *fn ft fm* from disk [RC=31|55|70|76|99|100+*nn*]
- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS414E Execid *execname exectype* already in storage [RC=1]
- DMS417E Only EXEC-2 and REXX EXECs are supported as storage resident EXECs [RC=4]
- DMS2521E EXECLOAD cannot be performed on empty file *fn ft* [RC=88]

The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

Reason	Location
Errors in the Shared File System	“File Pool Server Messages” on page 1414
Errors in using a file	“File Error Messages” on page 1418

EXECMAP



Notes:

[1] You can enter Options in any order between the parentheses.

Authorization

General User

Purpose

Use the EXECMAP command to list execs and XEDIT 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. For more information, see “SEGGEN” on page 868.

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.

exectype
is the type of the storage-resident exec(s) to be listed. If an asterisk (*) is coded in this field, all exectypes are used. The default is an asterisk.

If no operands are specified, the list contains all the execs and XEDIT macros in storage, the CMS installation saved segment, and loaded logical segments. Issuing EXECMAP with no operands is like issuing EXECMAP * *.

Options

User
indicates 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 *exectype* qualifications and that also have the USER attribute are listed. If neither USER, SYSTEM, nor SHARED is specified, all execs that satisfy the *execname* and *exectype* qualifications are listed. Shared execs are not listed if SET INSTSEG is OFF.

SYstem
indicates 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 *exectype* qualifications and that also have the SYSTEM attribute are listed. If neither USER, SYSTEM, nor SHARED is specified, all execs that satisfy the *execname* and *exectype* qualifications are listed. Shared execs are not listed if SET INSTSEG is OFF.

SHared
indicates the exec(s) is affected by the SET INSTSEG command. Only the execs that satisfy the *execname*, the *exectype*, or both qualifications and that also have the SHARED attribute are listed. If neither USER, SYSTEM, nor SHARED is specified, all execs that satisfy the *execname* and *exectype* qualifications are listed. Shared execs are not listed if SET INSTSEG is OFF.

SEGment
indicates only execs in the specified segment are to be listed. If neither SEGMENT or NOSEGMENT is specified, all execs will be listed, even if 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 *exectype* and reside in a logical segment.

NOSEGment
indicates only execs that do not reside in a logical segment are to be listed. If neither SEGMENT nor NOSEGMENT is specified, all execs will be listed, even if they reside in a segment.

STACK FIFO
STACK LIFO
specifies 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 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 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.
3. In an XA or XC mode virtual machine, the EXECMAP command operates on execs and macros loaded above and below the 16MB line with the EXECLOAD command.

4. When an EXEC or XEDIT macro has been EXECLOADed into storage and the EXEC or XEDIT macro is invoked through a CMSCALL on which a FBLOCK is supplied, the high–order bit of the FBLOCK address must be set on in order for the usage count reported by EXECMAP to be incremented. For more information on File Block, see *z/VM: REXX/VM Reference*.

Examples

To list all storage-resident execs with an exectype of XEDIT, 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
  .         .         .      .        .      .
```

Note:

1. The storage used (specified in the `Bytes` column) represents the amount of virtual storage required to maintain the exec in storage.
2. The usage count (specified in the `Usage` column) indicates the number of times the EXEC or XEDIT macro has been executed during the period of time that is has been EXECLOADed into storage.

Messages and Return Codes

- DMS070E Invalid parameter *parameter* [RC=24]
- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS416W There are no *execname exectype* {SYSTEM|[or]USER|[or]SHARED} EXECs storage resident [RC=28]
- DMS1284T 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 their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

EXECOS

Authorization

General User

Purpose

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.

Operands

cms_cmd
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 can indicate 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 REXX 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) that contain OS programs
   - OS environment flags (OSSFLAGS)

   All OS storage associated with an SVC level is released upon completion of the SVC level. If STORECLR = ENDSVC, EXECOS does not perform OS cleanup.

   If VSAM is running, VSAM cleanup is also done regardless of the setting of SET STORECLR.

   Explicit clearing of MACLIB and TXTLIB pointers and directories in STORCLR = ENDSVC mode can be accomplished by using the GLOBAL command, either within the EXECOS call or in your normal exec processing. GLOBAL MACLIB or GLOBAL TXTLIB without any name list will free the associated pointers and storage for any globaled libraries.

Note: Explicit freeing of these libraries may cause degraded application performance if many accessed disk members must be searched to reload the library members again for any subsequent usage. This is especially true if there are large OS disks in the search order that contain many members to search.

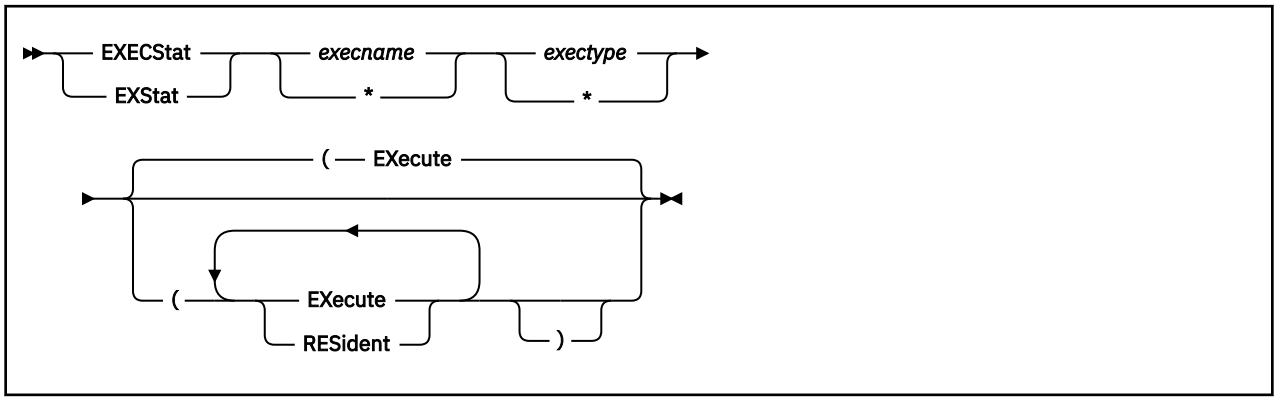
3. When processing in the VSAM environment, issue EXECOS prior to issuing CMS commands or user applications that run in the user area. Failure to do so could result in unpredictable errors and the overlay of VSAM control blocks.
4. When you request a reset of the OS environment after the execution of a CMS exec, the EXECOS command should *precede* the CMS exec command. For example:

```
&TRACE ALL
EXECOS EXEC VMFASM DMSSEB DMSVM
&EXIT
```

5. Whenever the CMS command, CP, must be used within an exec or EXEC 2 exec to transmit a CP command to the z/VM control program environment, a request for a reset of the OS environment after the execution of the CP command requires the EXECOS command precede the CMS CP command. For example:

```
&TRACE ALL
EXECOS CP QUERY DASD
&EXIT
```

EXECSTAT



Authorization

General User

Purpose

Use the EXECSTAT command to verify the existence of an exec or XEDIT macro in storage, on a disk, or in an SFS directory; or to determine where it will be executed from. The status is returned as a return code.

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.

Options

Note: These two options are mutually exclusive. If both are entered, the last one specified will be used.

EXecute
determines whether the EXEC will execute from storage or disk. This is the default.

RESident
verifies the existence of the EXEC in storage.

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.
3. In an XA or XC virtual machine, the EXECSTAT command operates on execs and macros loaded above and below the 16MB line with the EXECLOAD command.

Examples

To see if there are any storage-resident execs with the execname FILELIST, enter:

```
execstat filelist *  (resident
```

To see if there are any storage-resident execs with an exectype of XEDIT, enter:

```
execstat * XEDIT (resident
```

To see where the FILELIST EXEC will execute from, you would enter:

```
execstat filelist exec (execute
```

or

```
execstat filelist exec
```

Responses

The return codes and their meanings depend on the option specified.

With the EXECUTE Option:

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.

With the RESident Option:

Code
Meaning

0
Exec exists in storage. Register 1 contains pointer to the fileblock.

4
Exec exists in storage and there is another one on dasd that has a higher priority. Register 1 contains a pointer to the fileblock.

8
Exec exists in storage but is inactive (INSTSEG is OFF).

12
Exec does not exist in storage.

Messages and Return Codes

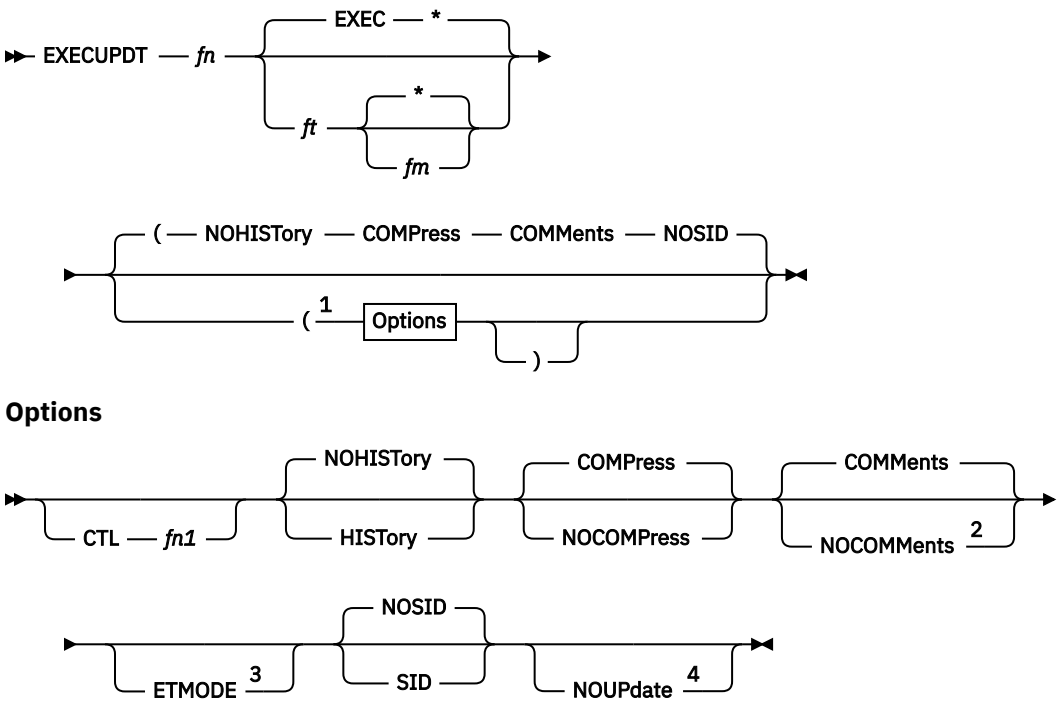
- DMS003E Invalid option: *option* [RC=24]
- DMS042E No execid specified [RC=24]

- DMS054E Incomplete execid specified [RC=24]
- DMS070E Invalid parameter *parameter* [RC=24]
- DMS109S Virtual storage capacity exceeded [RC=104]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

EXECUPDT



Notes:

[1] You can enter Options in any order between the parentheses.
[2] Do not specify NOCOMMENTS with COMPRESS.
[3] ETMODE only functions with NOCOMMENTS.
[4] HISTORY is ignored if NOUPDATE is specified.

Authorization

General User

Purpose

Use the EXECUPDT command to apply updates to a source program and create an executable version of the program. For more information on how to set up a program for updating, see *z/VM: CMS Application Development Guide* and for more information about editing files in update mode with XEDIT, see *z/VM: XEDIT Commands and Macros Reference*.

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.

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. It cannot be an empty file. If the file type or file mode is omitted, EXEC and * are assumed, respectively.

Options

CTL
specifies 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. It cannot be an empty file.

NOHISTory
specifies the UPDATES file created by the UPDATE command should not be appended to the updated source file. This is the default.

HISTory
specifies a file named ‘fn’ with a file type of UPDATES is to be appended to the updated source file as a comment. This file, ‘fn UPDATES’, is created by the UPDATE command.

NOCOMPress
specifies comments are not to be removed from the updated source file.

COMPress
specifies 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.

COMMents
specifies comments are not to be removed from the source file except those removed by the COMPRESS option. This is the default.

NOCOMMents
specifies all comments, leading blanks, and blank lines are to be removed from the source file. For REXX programs, one REXX comment line containing the exec name and file type is inserted at the beginning of the file.

ETMODE
specifies the source file contains DBCS characters and that shift-in and shift-out characters should be paired while comments are being removed.

SID
specifies the last 17 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 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 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.

Usage Notes

1. You may also use any other options accepted by the UPDATE command. For more information, see “UPDATE” on page 1092.
2. If you want to issue EXECUPDT from an exec program, you should precede it with the EXEC command; that is, specify

```
exec execupdt
```

3. If you use NOCOMMENTS on an exec that uses data stored within comments, the updated file may not execute properly. Also, if the exec uses comments that delimit a REXX function and cause the function to span more than one line, the updated file may not execute properly. For example, these lines:

   ```
   /*   */
   SAY DATE/* comments...
       .... */(USA)
   ```

   become, after EXECUPDT with the NOCOMM option:

   ```
   /*  fn EXEC  */
   SAY DATE
   (USA)
   ```

   EXECUPDT will not SPLIT/JOIN lines after removing comments.

Examples

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 a file named myfile xedit is produced from myfile $xedit without the comments.

Responses

If the update file being applied is an empty file, you get the following error message:

```
DMS178I  Applying fn ft fm (empty file)
```

If an AUX file is empty, you receive the following message:

```
DMS1229I  fn ft fm is empty
```

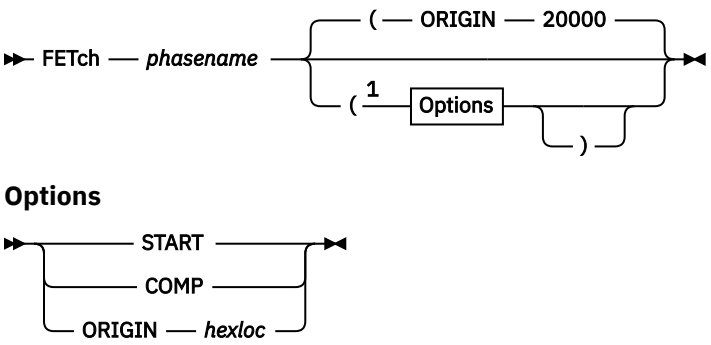
Processing continues.

Messages and Return Codes

- DMS001E No filename specified [RC=24]
- DMS002E File *fn ft fm* not found [RC=28]
- DMS003E Invalid option: *option* [RC=24]
- DMS007E File *fn ft fm* is not fixed, 80-character records [RC=32]
- DMS010W Premature EOF on file *fn ft fm*--sequence number *seqno* not found [RC=12]
- DMS024E File *fn ft fm* already exists [RC=28]
- DMS048E Invalid filemode *mode* [RC=24]
- DMS054E Incomplete fileid specified [RC=24]
- DMS065E *option* option specified twice [RC=24]
- DMS066E *option1* and *option2* are conflicting options [RC=24]
- DMS069E Filemode *mode* not accessed [RC=36]
- DMS070E Invalid parameter *parameter* [RC=24]
- DMS104S Error *nn* reading file *fn ft fm* from disk or directory [RC=100]
- DMS105S Error *nn* writing file *fn ft fm* on disk or directory [RC=100]

- DMS174W Sequence error introduced in output file: *seqno1* to *seqno2 seqno1* to *seqno2* [RC=8]
- DMS176W Sequencing overflow following sequence number *seqno* [RC=8]
- DMS179E Missing or invalid MACS card in control file *fn ft fm* [RC=32]
- DMS181E No update files were found [RC=40]
- DMS182W Sequence increment is zero [RC=8]
- DMS183E Invalid {CONTROL|AUX} file control card [RC=32]
- DMS184W ./S not first card in update file--ignored [RC=12]
- DMS185W Invalid character in sequence field *seqno* [RC=12]
- DMS186W Sequence number *seqno* not found [RC=12]
- DMS187E Option STK invalid without CTL [RC=24]
- DMS207W Invalid update file control card [RC=12]
- DMS210W Input file sequence error: *seqno1* to *seqno2* [RC=4]
- DMS299E Insufficient storage to complete update [RC=41]
- DMS300E Insufficient storage to begin update [RC=41]
- DMS361E Filemode *mode* is not a CMS disk or directory [RC=36]
- DMS419E *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]
- DMS637E Missing value for the CTL option [RC=24]
- DMS649E Extraneous parameter *parameter* [RC=24]
- DMS653E Error executing *command* rc=*nn* [RC=40]
- DMS671E Error updating *fn ft fm*; rc=*nn* from XEDIT [RC=100]
- DMS1229E *fn ft fm* is empty [RC=32]

FETCH



Notes:

[1] You can enter Options in any order between the parentheses.

Authorization

General User

Purpose

Use the FETCH command in CMS/DOS to load an executable phase into storage for execution.

Operands

phasename
is the name of the phase to be loaded into virtual storage. CMS searches for the phase:

1. In a VSE private core image library, if IJSYSCL has been defined
2. In CMS DOSLIBs that have been identified with the GLOBAL command
3. 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. For more information, see Usage Note “5” on page 261.

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. For more information, see Usage Note “6” on page 261.

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:

   ```
   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

   ```
   DMS740I  Execution begins ...
   ```

   message is not displayed.
6. The ORIGIN option is used by the IKQSAVE 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 BGCOM reflects only the ending address of the longest phase loaded, not that of the phase in the library that has the highest ending address.

Examples

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

```
DMS710I  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.

```
DMS740I  Execution begins ...
```

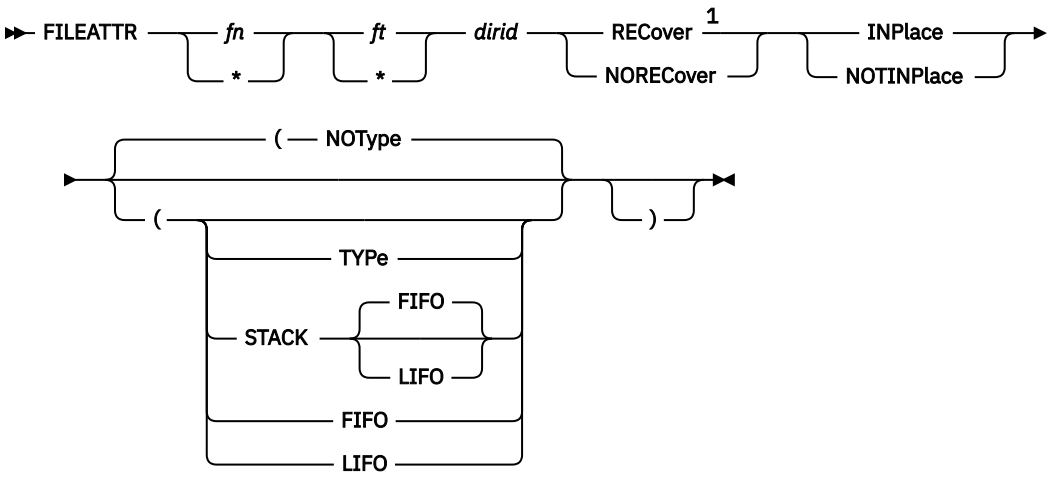
This message is issued when the START option is specified; it indicates program execution has begun.

Messages and Return Codes

- DMS003E Invalid option: *option* [RC=24]
- DMS004E Phase *phase* not found [RC=28]
- DMS016E No private core image library found
- DMS029E Invalid parameter *parameter* in the option *option* field [RC=24]
- DMS070E Invalid parameter *parameter* [RC=24]
- DMS098E No phase name specified [RC=24]

- DMS099E CMS/DOS environment not active [RC=40]
- DMS104S Error *nn* reading file *fn ft fm* from disk or directory [RC=100]
- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS113S Disk(*vdev*) not attached [RC=100]
- DMS115E Phase load point less than *vstor* [RC=40]
- DMS411S Input error code *nn* on SYS*aaa* [RC=*rc*]
- DMS623S Phase cannot be loaded at location *hexloc*--this area is available for system use only [RC=88]
- DMS623S Phase cannot be loaded at location *hexloc*--this area is available for system use only [RC=88]
- DMS777S DOS partition too small to accommodate fetch request [RC=104]
- DMS2521E FETCH cannot be performed on empty file *fn* [RC=88]

FILEATTR



Notes:

[1] RECOVER and INPLACE are conflicting operands.

Authorization

General User

Purpose

Use the FILEATTR command to modify the recoverability and overwrite attributes of file(s) residing in an SFS directory.

Operands

fn ft
specify the file name and file type of the SFS file for which you want to change the file attributes.

***
use when you want to change the extended file attributes of a group of SFS files.

dirid
is the directory identifier being searched. You can also use a file mode for the *dirid* if the directory is already accessed. For more information on *dirid*, see "Naming Shared File System (SFS) Directories" on page 6.

RECover
specifies the file is to be recoverable. When a file is recoverable, uncommitted changes are backed out as the result of an application initiated rollback.

NORECover
specifies changes to the file are not rolled back, in the event of an application initiated rollback. In most cases, the updates will be committed.

NOTINPlace
specifies the file writes are to be shadowed such that readers see a consistent version of the file from Open to Close, and will not see uncommitted updates.

INPlace
specifies updates are to be made in place where possible for reduced DASD utilization. Users that have an INPLACE file open for read have to re-open the file to see extensions (new blocks) that have

been written and committed to the file. Readers may not see a consistent version of the file from Open to Close.

Options

Type
displays information at the terminal.

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

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, 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. If FILEATTR is issued from an exec or assembler program and the specified file pool is active on the current default work unit (that is, has some work on it that has not been committed or rolled back), the command will fail.
2. You must have write authority to the specified file to change its attributes.
3. The user cannot change the attributes of a file in an SFS directory if the file is locked for:
   - SHARE, EXCLUSIVE, or UPDATE by another user
   - SHARE by you
4. Both overwrite (INPLACE/NOTINPLACE) and recoverability (RECOVER/NORECOVER) attributes must be specified.
5. FILEATTR can only be used to change the attributes of base files and aliases. If special characters are used to specify the file name or file type, or both, processing continues for the remaining file IDs that match the specified pattern.
6. You can invoke the FILEATTR command from the command line, from an exec, or as a function from a program. No error messages are issued if FILEATTR 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 REXX exec with ADDRESS COMMAND in effect

Messages and Return Codes

- DMS002E File *fn ft fm*|*dirname* not found [RC=28]
- DMS037E Base file for *fn ft fm* is in a DIRCONTROL directory accessed read/only [RC=36]
- DMS069E File mode *mode* is not accessed [RC=36]

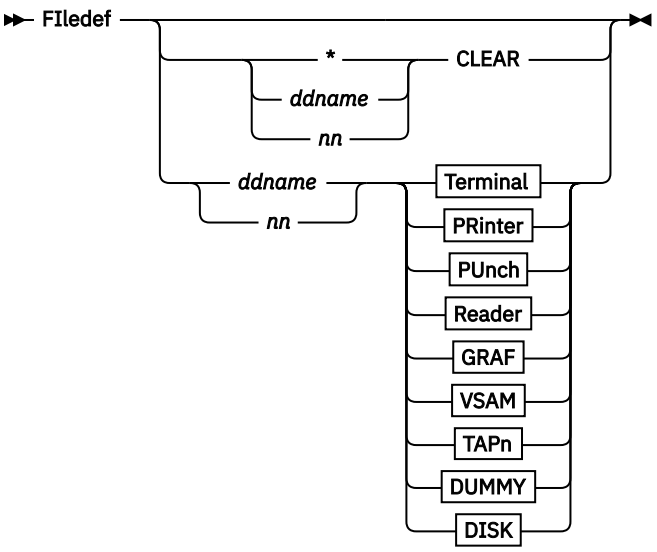
- DMS109S Virtual storage capacity exceeded. [RC=104]
- DMS1187E Too many subdirectory levels in *dirname* [RC=24]
- DMS1188E File mode *mode* is not associated with a directory. [RC=74]
- DMS1223E There is no default file pool currently defined. [RC=40]
- DMS1163E FILEATTR command failed for *fn ft fm*|*dirname* [RC=nn]
- DMS1184E File *fn ft fm*|*dirname* not found or you are not authorized for it [RC=28]
- DMS1291E There are no unused workunits available [RC=88]
- DMS2023E File pool *filepoolid* does not support the requested FILEATTR command [RC=88]
- DMS2024W File *fn ft fm*|*dirname* already has attributes: xxx and xxx [RC=4]
- DMS2025E RECOVER and INPLACE are conflicting file attributes. [RC=24]
- DMS2040E FILEATTR cannot be performed on a directory control directory that is accessed read-only [RC=36]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

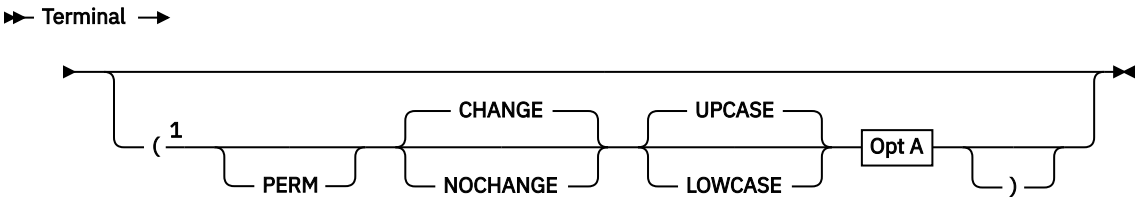
Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411
Errors in the Shared File System	“File Pool Server Messages” on page 1414

These error messages may be returned by the FILEATTR command:

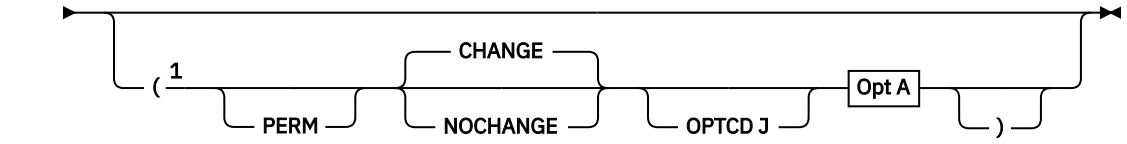
FILEDEF



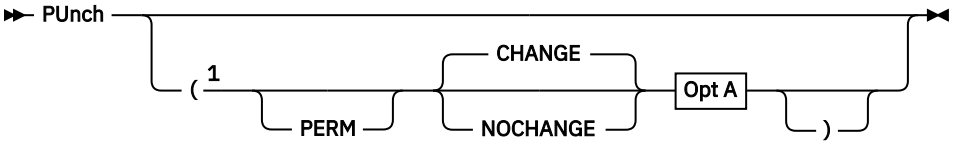
Terminal



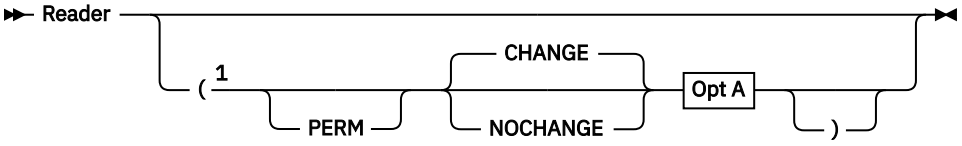
PRinter



PUnch



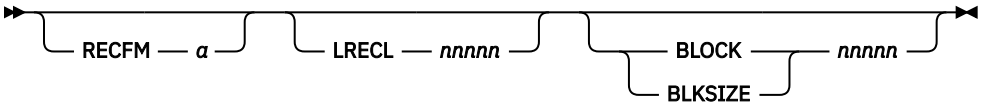
Reader



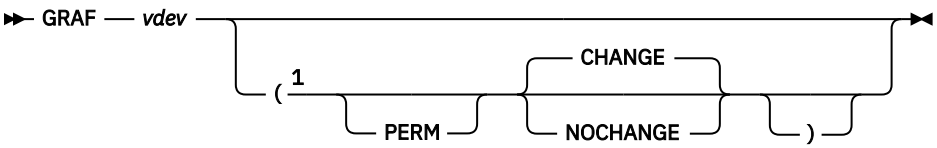
Notes:

[1] You can enter Options in any order between the parentheses.

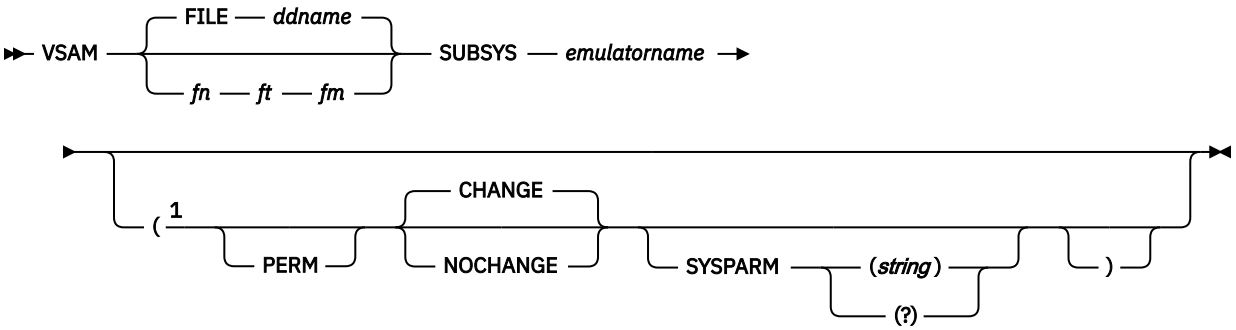
Opt A



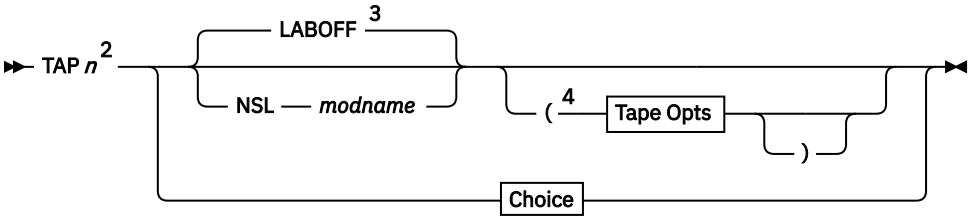
GRAF



VSAM



TAPn



Notes:

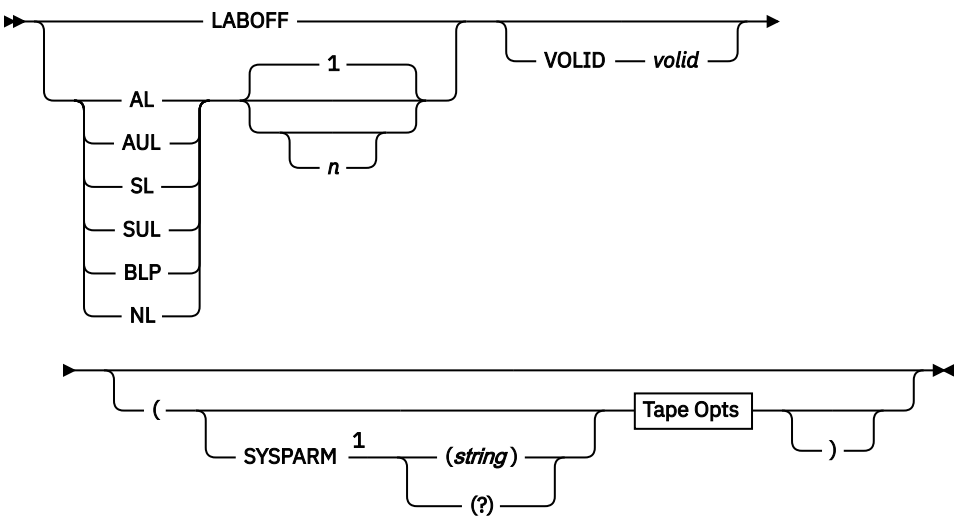
[1] You can enter Options in any order between the parentheses.

[2] If the *n* in TAP*n* is not specified, FILEDEF uses the device currently associated with the specified *ddname*. If there is no existing definition for the specified *ddname* or the device currently associated with it is not a tape device, FILEDEF defaults to TAP2.

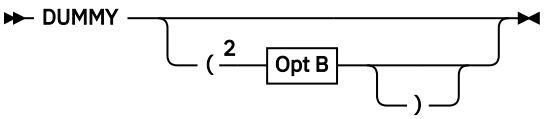
[3] This is the common CMS default label type.

[4] You can enter Options in any order between the parentheses.

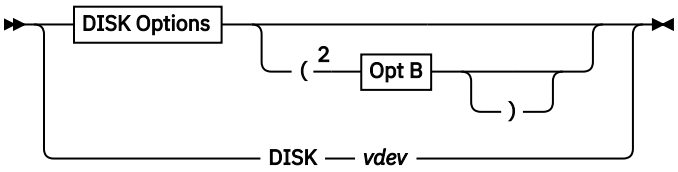
Choice



DUMMY



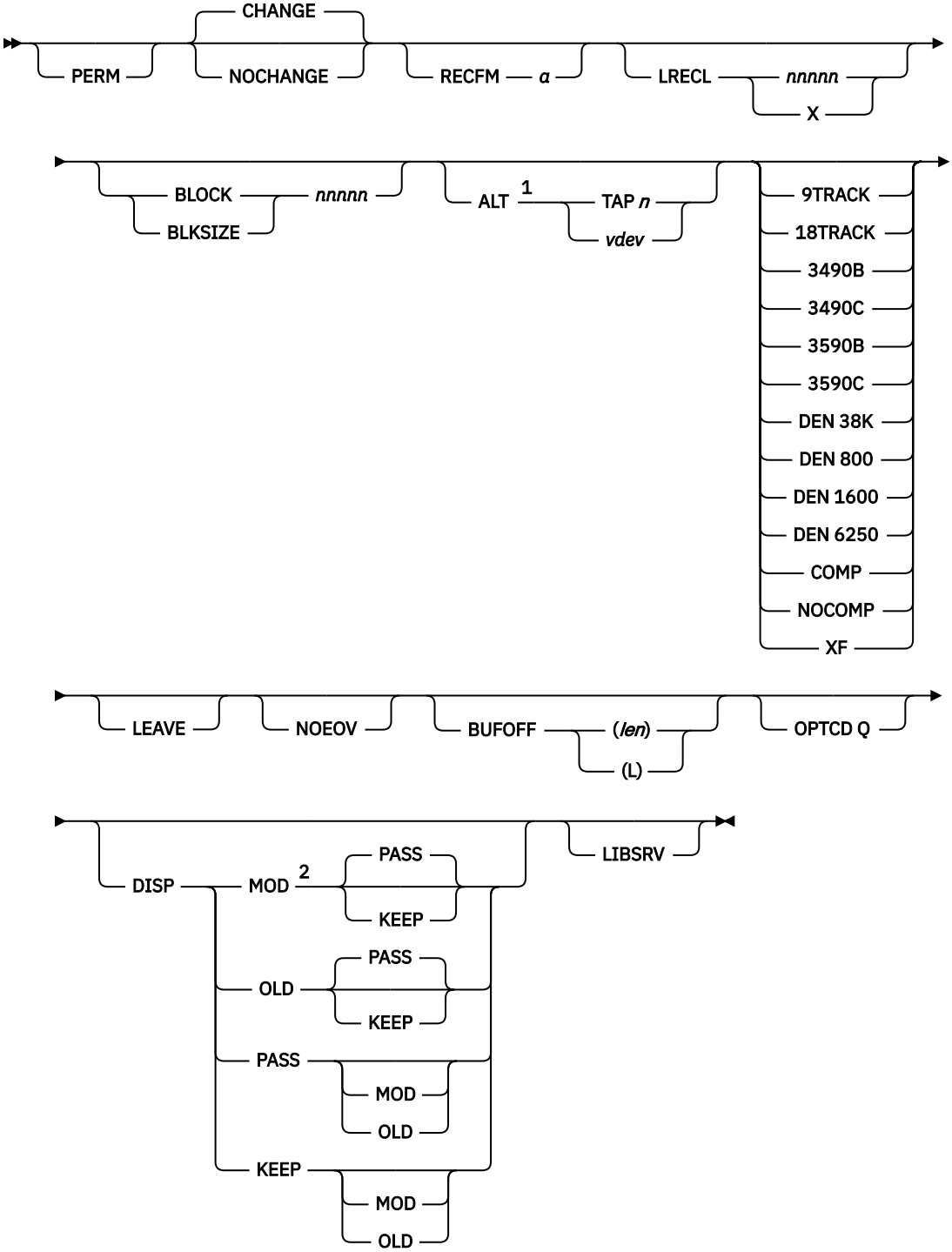
DISK



Notes:

[1] The SYSPARM option is valid only when a DMSTVI module is available.
[2] You can enter Options in any order between the parentheses.

Tape Opts

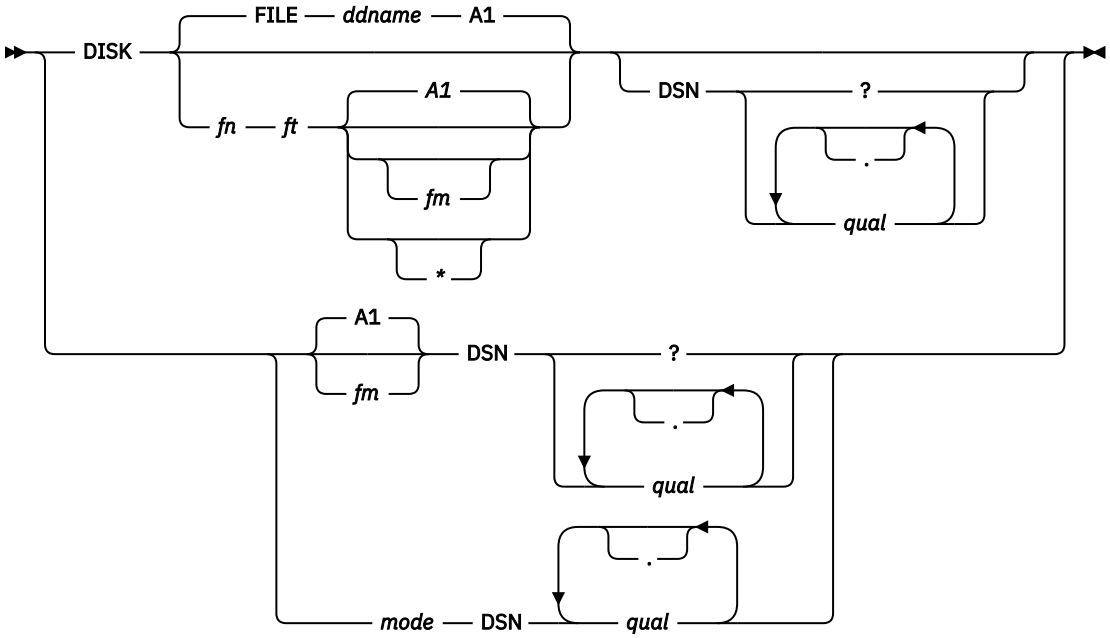


Notes:

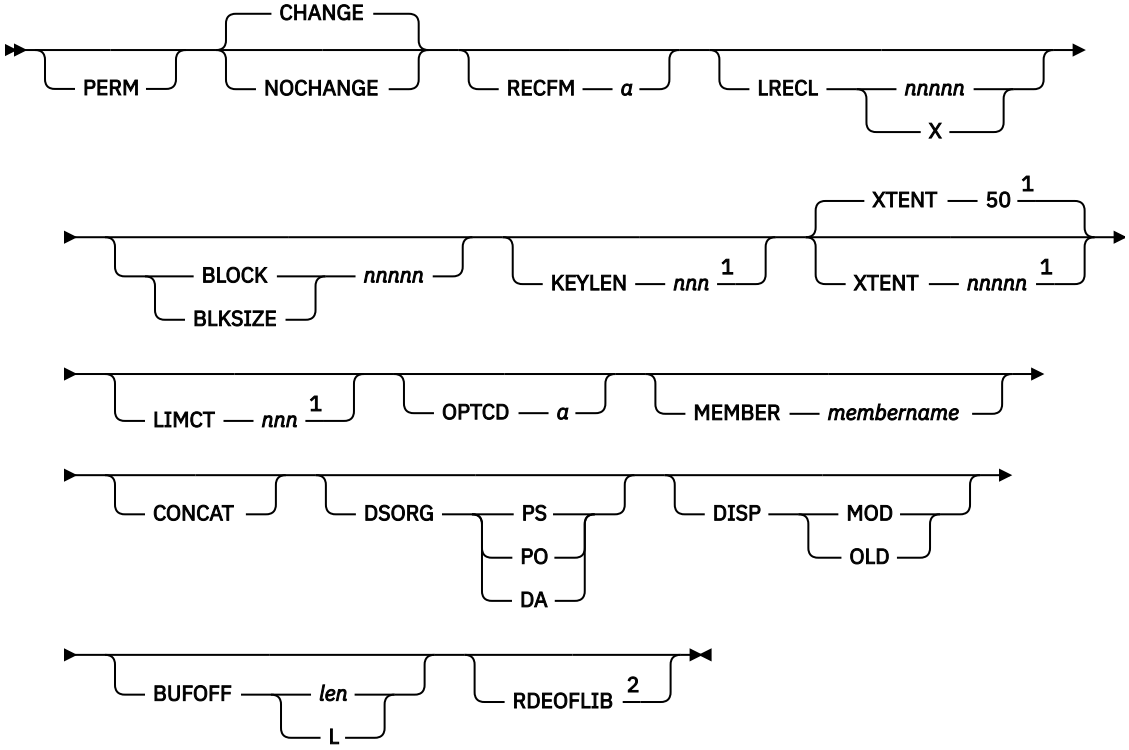
[1] ALT is valid only for SL, SUL AL, and AUL tapes.

[2] DISP MOD is valid only for SL or SUL tapes. DISP OLD, DISP PASS, or DISP KEEP is valid for all tapes.

DISK Options



Opt B



Notes:

[1] The KEYLEN, XTENT, and LIMCT options are meaningful for BDAM only.
[2] The RDEOFLIB option is meaningful only for CMS library file types (TXTLIB, MACLIB, DOSLIB, or LOADLIB).

Authorization

General User

Purpose

Use the FILEDEF command to establish data definitions for OS application *ddname*s, 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.

Operands

ddname
nn
****
: is the name by which the file is referred to in your program. The *ddname* can be 1 to 8 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 all file definitions not entered with the PERM option should be cleared.

CLEAR
: removes any existing definition for the specified *ddname*. Clearing a *ddname* before defining it ensures a file definition does not exist and that any options previously defined with the *ddname* no longer have effect.

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. For more information on the record length that is written using this operand, see Using the FILEDEF Printer Operand.

PUnch
: is the spooled punch.

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

DISK
: specifies the virtual I/O device is a disk or SFS directory. For more information on how to use the DISK operand, see Using the FILEDEF DISK Operand.

DUMMY
: indicates no real I/O takes place for a data set.

TAP*n*
: is a tape device.

  TAP*n* is one of the following device names for a virtual tape device: TAP0, TAP1, TAP2, TAP3, TAP4, TAP5, TAP6, TAP7, TAP8, TAP9, TAPA, TAPB, TAPC, TAPD, TAPE, TAPF. For more information on device names and virtual device numbers for tape devices, see *z/VM: CMS User's Guide*. You can also specify the type of label processing you want on your tape. For more information on specifying label processing, see Using the FILEDEF TAPn Operand.

GRAF
: specifies the virtual I/O device is a Graphic Display.

vdev
: is the virtual device number of the attached graphic display. The valid numbers are X'0000' through X'FFFF'.

VSAM
: specifies the file being defined is a VSAM file.

fn
: is the file name of the VSAM or DISK file being defined. If a file name is omitted, the default file name is FILE.

ft
is the file type of the VSAM or DISK file being defined. If a file type is omitted, the default file type is the *ddname*.

fm
is the file mode of the DISK file being defined. If a file mode is omitted, the default file mode is A1. A file mode of * indicates the file may be found on any disk in the search order of accessed disks. True OS files, which have the structure and size limits required for processing under a z/OS® system, have a file mode number of 4, indicating an OS structured file type.

SUBSYS *emulatorname*
specifies the name of the VSAM emulator that will process the VSAM function requests associated with this file. The emulator must either already exist as a nucleus extension or be available as a module or a member of a loadlib for FILEDEF to NUCXLOAD the emulator. The module or the loadlib and the loadlib member must have the same name as *emulatorname*.

Note: If the emulator is not already loaded, it is loaded as a nucleus extension with the service attribute. This means the emulator is invoked by CMS during NUCXDROP of the emulator during abend processing. The emulator must be prepared to handle this invocation.

Options

Whenever a non-valid option is specified for a particular device type, an error message is issued. The valid options for each device type are shown in the syntax diagram for the particular device type.

RECFM *a*
specifies the record format of the file, where *a* can be one of these:

a
Meaning

D
Variable length ASCII records

DB
Variable blocked ASCII records

F
Fixed length

FB
Fixed blocked

V
Variable length

VB
Variable blocked

U
Undefined

DS, DBS
Variable length, spanned ASCII records

FS, FBS
Fixed length, standard blocks

VS, VBS
Variable length, spanned records

These values **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.

A
ASA control characters

M
Machine codes

Note:

1. The **A** and **M** are not valid with ASCII **D**, **DB**, **DS**, and **DBS** record formats.
2. **D**, **DB**, **DS**, and **DBS** are valid only for ANSI labeled tapes.
3. **FB** and **VB** should not be used with TERMINAL or READER devices.
4. **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.
5. **M** should not be used with TERMINAL devices.

LRECL *nnnnn*
LRECL X
specifies the logical record length (*nnnnn*) of the file, in bytes. An LRECL of X indicates an OS variable spanned record size of 32768 (equal to 32K or X'8000'). When LRECL value of either 32768 or X is indicated, it is assumed Extended Logical Record Interface (XLRI) processing will be used to handle very large records as segment pieces up to an unknown maximum record length. For true OS Simulation files, because of OS restrictions, LRECL should not exceed 32760 bytes for fixed-length or VSAM records, or 32756 bytes for variable-length records. The Logical Record Interface (LRI), or segment oriented Extended Logical Record Interface (XLRI) allows variable spanned OS records to be larger than 32760 in length. VM allows LRI records to range from 5 to 65535 bytes in length. XLRI records have no upward size limit because they are handled as segment pieces. However, each spanned record segment piece can be no larger than the BLKSIZE-4 bytes. Note that OS variable record types include a 4-byte record length field as part of the LRECL. When CMS file mode number 4 is designated for an OS variable file, the 4-byte record length field is stored as part of the data simulating an OS record structure format. Non-OS CMS files do not have these length values as part of the data and may have LRECL sizes up to 65535 bytes for both fixed and variable records.

BLOCK *nnnnn*
BLKSIZE *nnnnn*
specifies the logical block size (*nnnnn*) of the file, in bytes. For true OS Simulation files, because of OS restrictions, this value should not exceed 32760. Note that OS variable blocked record types include a 4-byte block length field as part of the block size. When CMS file mode number 4 is designated for an OS variable file, the 4-byte block length field is stored as part of the data simulating an OS record structure format.

For non-OS CMS files, the concept of block size does not really exist, and the logical block size can be considered equivalent to the logical record length. Therefore, CMS files can have a block size specified up to 65535, as long as an LRECL is not also specified with a conflicting size.

If the BLOCK and BLKSIZE options are both specified, the value for BLOCK is used and BLKSIZE is ignored. (For convenience in the rest of this command description, BLKSIZE refers to either option.) If BLKSIZE and LRECL are both specified and the BLKSIZE is larger than the LRECL, this implies the data to be handled may be blocked, a true OS file convention. For fixed- length records, the BLKSIZE must be an even multiple of the LRECL, and the RECFM must be fixed-block (FB). For variable-length records, you can specify the BLKSIZE larger than one record length. If variable blocking is required, a RECFM of VB should be specified. For variable spanned records, the BLKSIZE can be less than the size of one record. The record data is split up and spanned across multiple blocks. This allows the BLKSIZE to be tailored to best fit the device for either performance or storage capacity.

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, and so forth) of the new FILEDEF command if a file definition exists for the specified *ddname*.

SYSPARM
passes the address of the character value *string* to DMSTVI (a system interface routine supplied for tape volume handling. For more information on DMSTVI, see *z/VM: CMS Application Development Guide for Assembler*. The use and meaning of *string* are determined by your DMSTVI module. For VSAM, the string is passed to the VSAM emulator when the file is opened. The maximum length of the string is determined by the total length of the command.

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 numbering
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 the output line contains a TRC (Table Reference Character) byte.

OPTCD Q
is valid only for the TAP*n* option. On input from a tape, this option specifies the information be translated from ASCII to EBCDIC. On output, it specifies the information be translated from EBCDIC to ASCII. Translation is supported for AL, AUL, LABOFF, BLP, and NL tape label processing, and for record formats of F, FB, FS, FBS, D, DB, DS, and DBS.

Note: The KEYLEN, XTENT, LIMCT, and OPTCD options should only be used with BDAM, QSAM, or BSAM files.

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 normally only be used to read members.

RDEOFLIB
allows a disk CMS library of file type TXTLIB, MACLIB, DOSLIB, or LOADLIB to be read until physical end of file, ignoring member boundaries. This option is mutually exclusive with the MEMBER option.

CONCAT
allows you to assign the same *ddname* to two or more OS libraries so you can refer to them in a single GLOBAL command. You may concatenate libraries of any file type, including macro libraries of file type MACLIB, text libraries of file type TXTLIB, and load libraries of file type LOADLIB. For more information on the CONCAT option, see Usage Notes for the CONCAT Option.

DSORG PS
DSORG PO
DSORG DA
is the data set organization: physical sequential (PS), partitioned (PO), or direct access (DA).

DISP OLD
positions the read/write pointer. Input files will be positioned at the first record. If a DISK file is opened with any output intent, file update is assumed. The write pointer will be positioned after the last record when the CMS file system opens the file, but OPEN will reset to the first record when OPEN processing is done. DISK files will therefore maintain their current file contents if no actual I/O occurs between OPEN and CLOSE. Tape files will always be positioned at the first record. If the tape is labeled and is an output type, the tape labels will be rewritten during OPEN and CLOSE macro processing, causing the file to be updated even if no data I/O occurs.

Note: DISP OLD can be specified with **KEEP** or **PASS**.

DISP MOD
positions the read/write pointer. This option should only be used for adding records to the end of a file. If TAP*n* is specified, the label type must be SL or SUL, for IBM Standard Label tapes. For DISK, 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 multivolume tape processing this is valid only with the currently mounted volume; no volume switching is done.

Note: DISP MOD can be specified with **KEEP** or **PASS**.

DISP KEEP
rewinds the tape when the CLOSE macro is issued if no positioning is specified on the CLOSE macro. Any tape positioning parameter specified on the CLOSE macro itself will override the DISP KEEP specification in the FILEDEF command. Specifying DISP KEEP on the FILEDEF command and no positioning on the CLOSE macro is equivalent to specifying REWIND positioning on the CLOSE macro.

DISP PASS
positions the tape at the logical end of file on normal CLOSE or at the logical end of data on CLOSE TYPE=T if no positioning is specified on the CLOSE macro. Any tape positioning parameter specified on the CLOSE macro itself will override the DISP PASS specification in the FILEDEF command. Specifying DISP PASS on the FILEDEF command and no positioning on the CLOSE macro is equivalent to specifying LEAVE positioning on the CLOSE macro.

LIBSRV
places the tape drive under the control of a Tape Library Dataserver machine. If the DFSMS/VM Removable Media Services (RMS) FSMPPSI CSLLIB is available to CMS, requests for new tape mounts and unmounts through CMS native tape processing functions invoke calls to RMS CSL routines.

Note: If a pre-attached tape drive is found that cannot be controlled by RMS commands, the LIBSRV option is ignored during mount or unmount processing.

3490C
specifies 3490 Data compaction mode (3490 only).

3490B
specifies 3490 Basic recording mode (3490 only).

3590C
specifies 3590 Data compaction mode (3590 only).

3590B
specifies 3590 Basic recording mode (3590 only).

XF
specifies 3480 data compaction mode.

9TRACK
specifies *any 9 track recording format*. CMS selects a 9 track recording format for you that the device is capable of writing (if there is one). If you require a particular 9 track recording format, use the DEN 800, DEN 1600, or DEN 6250 option.

Note: The TRACK and DEN options are not applicable for a 9346 cartridge tape unit. If you try to use any of these options, be aware no error messages will be issued at the time the option is specified; such errors will not be identified until an actual I/O operation, such as MOVEFILE, is attempted.

18TRACK
specifies 3480 Basic recording format.

DEN 38K
specifies 3480 Basic recording format.

DEN 800
specifies NRZI recording format.

DEN 6250
specifies GCR recording format.

DEN 1600
specifies PE recording format.

Note: The **DEN** option is not compatible with 9346 tapes.

COMP
specifies any data compaction available should be used. Since only the 3590, 3480, and 3490 IDRC drives support data compaction, this option only has meaning for those drives. CMS selects a compacted recording format for you that the device is capable of writing (if there is one). If you require a particular compacted recording format, use the 3590C, 3490C, or XF option.

NOCOMP
specifies data compaction is not wanted. The 3490 drive defaults to data compaction, but the 3480 does not. This option is intended for the 3480 and 3490 IDRC drives. CMS selects an uncompacted recording format for you that the device is capable of writing (if there is one). If you require a particular uncompacted recording format, use the 3490B, 3590B, 3490B, 18TRACK, DEN 6250, DEN 1600, or DEN 800 option.

LEAVE
is only valid for TAP*n* files that are IBM standard labels (SL or SUL), ANSI labels (AL or AUL), or when bypassing label processing (BLP). LEAVE determines the positioning of the tape during OPEN processing. If LEAVE is specified, the tape is not moved before label processing. If LEAVE is not specified, tapes with files specified as SL, SUL, AL, AUL, or BLP are rewound and then positioned before the files are processed.

NOEOV
is only valid for TAP*n* 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 IBM standard labeled tapes or ANSI labeled tapes, if NOEOV is specified, it is ignored during end-of-volume processing. For more information on end-of-volume processing, see *z/VM: CMS Application Development Guide for Assembler*.

ALT TAP*n*
ALT *vdev*
is only valid for TAP*n* files that are SL (standard label), SUL, AL (ANSI labeled) or AUL. This option specifies an alternate tape drive to be used when an EOV 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.

You specify the device name (TAP*n*), or alternatively the virtual device number (*vdev*) of the tape device which is to be the alternative device. The following names and corresponding virtual device

numbers are valid. For more information on the device names and virtual device numbers for tape devices, see *z/VM: CMS User's Guide*.

Device Name	Device Number	Device Name	Device Number
TAP0	0180	TAP8	0288
TAP1	0181	TAP9	0289
TAP2	0182	TAPA	028A
TAP3	0183	TAPB	028B
TAP4	0184	TAPC	028C
TAP5	0185	TAPD	028D
TAP6	0186	TAPE	028E
TAP7	0187	TAPF	028F

You may omit the leading zero on the device numbers.

A multivolume tape file must be recorded in the same recording format on each volume. This means that if you are *reading* tape files, both the primary and alternate tape devices must be capable of reading the recording format in which it is written. If you are *writing* tape files, both the primary and alternate tape devices must be capable of writing the recording format you select. If you do not select a specific recording format, the alternate device must be capable of writing the recording format CMS chooses as a default on the primary device. If your devices do not meet these criteria, you will get I/O errors on the second volume.

Note: Multivolume tape processing is not available with the 9346 cartridge tape unit.

UPCASE
translates all terminal input data to uppercase.

LOWCASE
retains all terminal input data as typed in.

BUFOFF(*len*)
specifies the length, in bytes, of the block prefix of an ASCII data set. Block prefixes are used with fixed, variable, and variable spanned ASCII records. For an input data set, the value can be 0 to 99 bytes or L. When writing to an output file, BUFOFF must be 0 or L. In CMS, you can only write a block prefix by using BUFOFF=L.

BUFOFF(L)
specifies the block prefix is four bytes long, contains the length of the block, and is treated as a block descriptor word.

Usage Notes

General Usage Information

1. If the FILEDEF command is entered with no operands, a list of current definitions is displayed. This is equivalent to issuing the QUERY FILEDEF command.
2. 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: The 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.

3. 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
   ```

   For information on the default file definitions made by the assembler, see "ASSEMBLE" on page 46.
4. For more information about the FILEDEF CLEAR Operand, see Using the FILEDEF CLEAR or CHANGE Operand.
5. CMS supports one virtual reader at number 00C, one virtual punch at number 00D, and one virtual printer at number 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 number indicated.

   If a program has two or more data control blocks (DCBs) with different *ddname*s open for the same unit record devices, records from different files may be mixed together into one file. Separate files are not maintained.
6. 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 any file format 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.
7. 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 instead of the FILEDEF command.
8. 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 more information on how to use this option of the FILEDEF command, see *z/VM: CMS Application Development Guide for Assembler*.
9. True OS Simulation files are defined as files that must adhere to the size limit and structure required for processing by a z/OS operating system. Any of the following file characteristics trigger CMS OS Simulation to treat the file as an OS type:
   - The TERMINAL, PRINTER, PUNCH, READER, GRAF, or VSAM operand is specified.
   - A tape standard label type is specified: SL, SUL, AL, or AUL.
   - The RECFM option is specified with either a blocking or spanning subspecification.
   - *ft* is specified using an OS pseudo-library keyword of LOADLIB, TXTLIB, MACLIB, or DOSLIB.
   - File mode number 4 is specified, or is already defined for the file on disk.
   - The DSN option is specified.
   - The KEYLEN, XTENT, LIMCT, or CONCAT option is specified.
   - The MEMBER option is specified with the name of an OS partitioned data set.
   - A DSORG option other than PS is specified.
   - The file is found to exist on an OS- or DOS-formatted disk.

Using the FILEDEF CLEAR or CHANGE Operand

1. Do not attempt to CLEAR or CHANGE a FILEDEF in a program while the corresponding DCB is open — you must do this before the DCB has been opened or after it has been closed. The OS macros rely on information set up by FILEDEF. If this information is cleared or changed, the macros are not able to function properly, and the results may be unpredictable.
2. The FILEDEF * CLEAR command should NOT be issued from an EXEC or called program. This command clears ALL the FILEDEFs (except for those defined with the PERM option), including those a previously called EXEC or program may have set up and still needs.

   For example:

   ```
   Exec A
    Filedef IN DISK A INPUT A
    LOAD PROGA (START
   ```

```
Exec B
 Filedef BFILE DISK B FILE A
        ..
        ..
 Filedef * Clear

PROGA
 OPEN IN
 READ IN...
 CHECK...
 Call ExecB   <-ExecB clears the filedefs
 READ IN...   <-unpredictable errors occur here
 CHECK...
 CLOSE IN     <-unpredictable errors would occur
                  here if the second Read
                  were not done
 IN  DCB DDNAME=INFILEA...
```

EXEC B should clear only the FILEDEF(s) which it defined. A program or EXEC can never be sure it was not called from another program or EXEC which defined other FILEDEFs. It could have been called from CMS Ready mode in a case where the user has done FILEDEF commands. A FILEDEF * CLEAR command will clear not just the FILEDEFs defined by one program or EXEC, but will clear ALL the non-permanent FILEDEFs. If OS simulation then tries to do any processing on DCBs requiring those FILEDEFs, unpredictable results may occur, ranging from doing I/O to the wrong file to taking a wild branch and abending CMS.

Using the FILEDEF Printer Operand

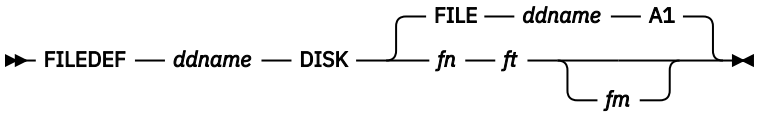
If you execute FILEDEF for a printer, the record length written to the virtual printer spool file will be the maximum record length of the spooled device as defined by CP. For example, if a printer is spooled as printer device type 4248, the record length written to the virtual printer will be 168 bytes. This is not affected by the LRECL option of FILEDEF and you cannot alter it. The maximum record length by device type is as follows:

Virtual Printer Type	Maximum Record Length
1403	132
3203	132
3800	204
4248	168
VAFP	32767

Using the FILEDEF DISK Operand

There are three general forms for specifying the DISK operand in a FILEDEF command.

1. Format 1



In this form, *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 a FILEDEF command is issued with no *fm* or with *fm* *, when OS simulation opens that file it will check every file on every disk in the search order until it finds the file. If a disk accessed before the disk where the file resides contains many files, the system will take significant time to check each fileid for a match. This slow response is particularly noticeable when *fm* * is specified and a real full-pack MVS™ disk is accessed before the desired disk. The MVS disk may contain many thousands of files and each dataset name is checked against the *fn* and *ft* requested. Where it is feasible, you should specify a definite *fm* on the FILEDEF command to avoid these large searches. When this is not possible, your

performance will be better if you make sure any disks containing large numbers of files are accessed later in the search order than the disk containing the desired file.

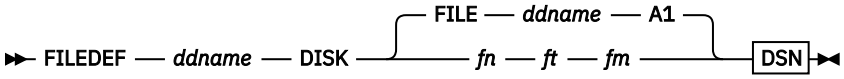
You can only use this form if the OS data set name or VSE file ID conforms to the OS naming convention (1-8 byte qualifiers separated by periods, to a maximum of 44 characters, including periods). Also, the data set name can have only two qualifiers; otherwise, you must use the DSN ? or DSN *qual1*... form. (See Format 2, below). If the OS data set name or VSE file ID is TEST.SAMPLE, you would enter:

```
FILEDEF MINE DISK TEST SAMPLE B1
```

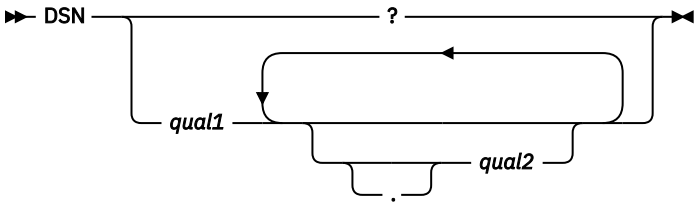
You can also use this form to concatenate TXTLIBs. However, the TXTLIBs must be specified on the GLOBAL command. For example:

```
GLOBAL TXTLIB TXLIB1 TXLIB2 TXLIB3
FILEDEF SYSLIB DISK TXLIB1 TXTLIB A (CONCAT
LKED fn
```

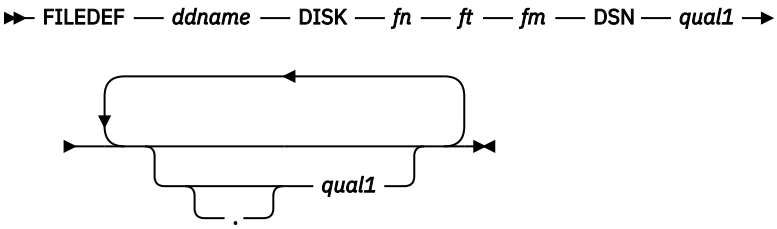
2. Format 2



DSN



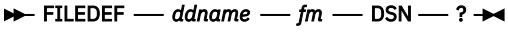
This form of the DISK operand is used only with OS data sets and VSE files. It allows you to enter OS and VSE file identifications that do not conform to OS data set naming conventions. The DSN operand corresponds to the DSN parameter on the OS DD (data definition) statement. There are many ways you can specify this form:



a.

The above form of the FILEDEF command associates 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.

b. 

c. 

These forms of the FILEDEF command allow you to specify the OS data set name or VSE file ID interactively, which allows you to enter names with special characters such as blanks, or consisting of more than two qualifiers. Type in DSN ?; CMS then requests the OS data set name or VSE file ID

exactly as it appears in the data set or file. For example, for an OS data set name or VSE file ID of TEST.SAMPLE.MAY, you would enter:

```
FILEDEF MINE DSN TEST SAMPLE MAY
```

or

```
FILEDEF MINE DSN TEST.SAMPLE.MAY
```

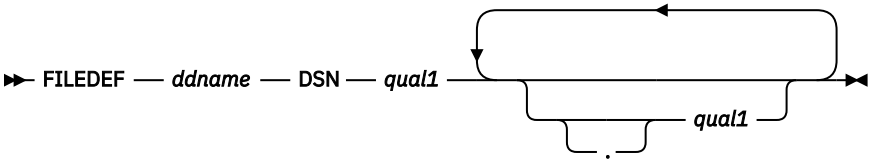
or

```
FILEDEF MINE DSN ?
```

and then

```
TEST.SAMPLE.MAY
```

When you use this form, you must code the periods separating the qualifiers. If you use this form, the default file name and file type for your file is the CMS file name and file type associated with the OS data set name or VSE file ID, and the default file mode is A1.



d.

With this form, you can explicitly specify the OS data set name or VSE file ID. 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 could enter:

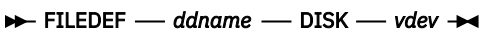
```
FILEDEF ddname B1 DSN MY FILE IN
```

or

```
FILEDEF ddname B1 DSN MY.FILE.IN
```

The default value for the file name and file type is FILE *ddname*.

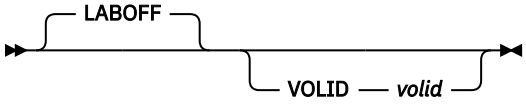
3. Format 3



This form associates a *ddname* with a virtual minidisk number. 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 *vdev* is the virtual number of the device referred to in your program by *ddname*.

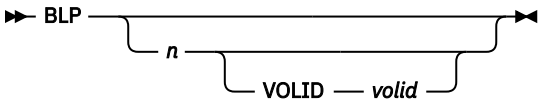
Using the FILEDEF TAPn Operand

Along with the TAP*n* operand, you can specify the type of label processing to be done for the tape file you are defining with FILEDEF.



1.

indicates no CMS tape label processing. This is the default. The tape is not positioned if this operand is specified.

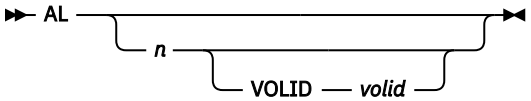
2. 

   indicates label processing is bypassed, but the tape is to be positioned before the file is processed.

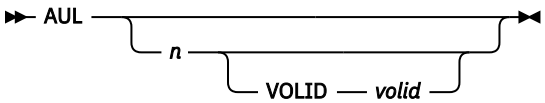
   Example

   ```
   filedef filea tap2 blp 2
   ```

   positions the tape to the second file, but does not check to see if the tape has a label.

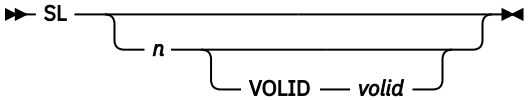
3. 

   indicates you are using ANSI labels.

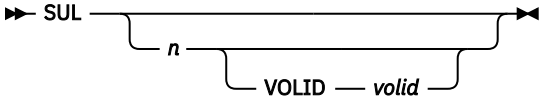
4. 

   indicates you are using ANSI user labels.

   Note: User label exits will not be processed for MOVEFILE.

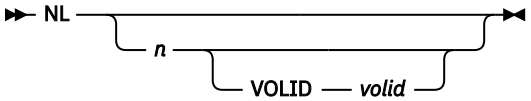
5. 

   indicates you are using IBM standard labels.

6. 

   indicates you are using standard user labels.

   Note: User label exits will not be processed for MOVEFILE.

7. 

   specifies your tape has no ANSI or IBM standard labels. (Do not use this operand if your tape has a VOL1 label. A file on it will not be opened.) If you mount a blank tape and specify NL, the tape will run off the end of the reel. Writing a tape mark will prevent this.

   Example 1

   ```
   filedef filea tap2 nl 2
   ```

   defines tap2 as having no labels. Where:

   n
   specifies the position of the file on a multifile volume. This can be used for the AL, AUL, SL, SUL, BLP, and NL files. If not specified, the default is 1.

   Example 2

```
filedef filea tap2 sl volid DEPT10
```

specifies the file on tap2 with standard labels. When this tape file is opened, CMS checks to see 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
```

Where:

volid
specifies a 1-6 character volume serial number to be verified by reading the VOL1 label on the tape. This can be used for the AL, AUL, SL, and SUL files. The valid *volid*s can be uppercase alphabetic, numeric, or these symbolic characters:

For AL tapes
" ; : < = > ? - . + / , * & % ! ( ) '

For SL tapes
¢ $ # - / @

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. VOLID is only valid for SL, SUL, AL, or AUL tape files. If VOLID is not specified, the volume label on the tape is not checked.

The CMS LABELDEF command also allows the user to set one or more *volid* values defining the limits of a dataset across multiple tape volumes. If a LABELDEF *volid* command is issued for the same *ddname* as an existing FILEDEF command, the first LABELDEF *volid* will replace the one specified with FILEDEF.

In multi-volume processing, for subsequent volumes, the actual *volid* from the tape mounted is used from the Volume Serial Number field in the volume label (VOL1) to record the multi-volume order. If no explicit LABELDEF was done, each new tape mounted has its serial number recorded as it is read. For output operations, the *volid* of the first volume in the set is written to the 'Data Set Serial Number' field in the HDR1 label, to identify the logical start of the data.

8. NSL *modname*

   indicates you are using nonstandard labels. Where:

   modname
   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 they start at an address that does not allow them to overlay a user program if they are to be used for NSL routines. For more information on the writing routines to process nonstandard labels, see *z/VM: CMS Application Development Guide for Assembler*.

   Example

   ```
   filedef filea tap2 nsl nonstd
   ```

   specifies the tap2 with nonstandard labels, by using the routine NONSTD.

   In multivolume processing, for subsequent volumes, the actual *volid* from the tape mounted is used in the Volume Serial Number field in the volume label (VOL1), and the *volid* of the first volume is written to the 'Data Set Serial Number' field in the HDR1 label.

9. In OS simulation, multivolume tape processing only applies to CMS OS QSAM simulation for IBM standard labeled (SL) and ANSI labeled (AL) tapes.

   In CMS/DOS simulation, multivolume tape processing is not supported. The multi or alternate volume parameters of FILEDEF or LABELDEF are not applicable.

In a CMS/DOS environment, it is not possible to create or process multivolume tapes. If a large file needs to be copied to tapes the COUNT and SKIP parameters of the IDCAMS REPRO function can be used to create multiple tapes where each tape contains only a segment of the file.

Usage Notes for Options LRECL, BLKSIZE, RECFM

1. LRECL and BLKSIZE values may be specified in either the DCB macro or the FILEDEF command. If the LRECL or BLKSIZE is specified in both the DCB and the FILEDEF command, the value from the DCB macro is used.
2. The LRECL and BLKSIZE values used by OS simulation are determined according to the file that is opened. When a file is opened and either the LRECL or the BLKSIZE is specified, the values are determined according to Table 13 on page 284.
3. If the open intent implies writing to (but not replacing) an existing minidisk or SFS directory file and the file is not a partitioned data set, the BLKSIZE is set to the actual record length of the CMS file, regardless of how it was previously specified. For this case, the LRECL can be determined according to the rules in Table 13 on page 284.

Table 13. Determining BLKSIZE and LRECL on CMS DASD when either LRECL or BLKSIZE, or both, are specified

BLKSIZE	LRECL	Results
Specified	Not specified	When simple RECFM values are used: • For RECFM=F: – LRECL=BLKSIZE • For RECFM=V|D|U: – LRECL=BLKSIZE-4 for true OS file types – LRECL=BLKSIZE for CMS file types
Specified	Not specified	When compound RECFM values are used (OS only): • For RECFM=FB|FBS: – LRECL=BLKSIZE • For RECFM=VB|VS|VBS|DB: – LRECL=BLKSIZE-4 • Under XLRI for RECFM=VS|VBS|DS|DBS: – LRECL=BLKSIZE-4 • Under LRI for RECFM=VS|VBS|DS|DBS: – LRECL=length of record area-32 or BLKSIZE-4
Not specified	Specified	When simple RECFM values are used: • For RECFM=F: – BLKSIZE=LRECL • For RECFM=V|D|U: – BLKSIZE=LRECL+4 for true OS file types – BLKSIZE=LRECL for CMS file types

Table 13. Determining BLKSIZE and LRECL on CMS DASD when either LRECL or BLKSIZE, or both, are specified (continued)

BLKSIZE	LRECL	Results
Not specified	Specified	When compound RECFM values are used (OS only): • For RECFM=FB|FS|FBS: – BLKSIZE=LRECL • For RECFM=VB|VS|VBS: – BLKSIZE=LRECL+4 or 32760, whichever is smaller
Specified	Specified	The specified values are used.

4. If neither the LRECL nor the BLKSIZE is specified and, (1) the file is opened for OUTPUT or OUTIN without DISP MOD (meaning the file will be replaced), or (2) the file being opened does not exist, message DMSSOP036E is issued with error code 4. This is because CMS cannot determine what LRECL and BLKSIZE to use. Otherwise, the rules in Table 14 on page 285 apply when neither the LRECL nor the BLKSIZE is specified.

*Table 14. Determining BLKSIZE and LRECL on CMS DASD when **neither** LRECL nor BLKSIZE is specified.*

BLKSIZE	LRECL	Results
Not specified	Not specified	• For RECFM=F|FB|FS|FBS: – BLKSIZE=CMS record length – LRECL=CMS record length • For RECFM=V|VB|VS|VBS|D|DB: – BLKSIZE=CMS record length – LRECL=CMS record length-4 for true OS file types – LRECL=CMS record length for CMS file types • Under LRI for RECFM=VS|VBS|DS|DBS: – BLKSIZE=CMS file record length or 32760 – LRECL=length of record area-32 or BLKSIZE-4

5. If FB is specified for the RECFM, the BLKSIZE must be an even multiple of the LRECL.
6. If D, DB, V, or VB is specified for the RECFM, the LRECL must be at least 4 bytes less than the BLKSIZE. Additionally, the LRECL must be at least 4 bytes greater than the largest user data record of the file.
7. If DS, DBS, VS, or VBS is specified for the RECFM, the LRECL can exceed the specified BLKSIZE, but the LRECL should not exceed a maximum value of 32756 because of normal OS restrictions. The Logical Record Interface (LRI) and the Extended Logical Record Interface (XLRI) will allow spanned LRECLs larger than 32756, but these large sizes may cause data interchange problems with other operating systems that cannot handle these large records.
8. Under the Logical Record Interface (LRI), if a record area is not present and DCB BFTEK=A is specified when a file is opened, CMS tries to define a record area. The record area size is the LRECL specified in the DCB or by FILEDEF plus 32 bytes. If the LRECL is not provided in the DCB or by FILEDEF, or in a tape standard label, a default value for the LRECL is first set to the BLKSIZE-4, if a BLKSIZE is provided. If neither an LRECL nor a BLKSIZE is provided, the LRECL is defaulted to 32760 and the record area size will default to a size 32 bytes larger than that.

   Exception: If LRECL to be used equals 32768 (LRECL=X), the DCB BFTEK=A interface will be ignored and segment oriented Extended Logical Record Interface (XLRI) processing will be used with simple

buffering to process the records without a record area. Simple buffers will be obtained as large as the BLKSIZE to process the segments.

Whenever possible, you should use a record area sized to the data you intend to process. You can specify the size of the record area using the BUILDRCD macro, or provide either the DCB or FILEDEF LRECL option.

Whenever possible, you should use a smaller record area than the default for better performance. You can specify the size of the record area using the DCB or BUILDRCD macro or the FILEDEF command.

9. On standard labeled tape, an LRECL of 99999 will be considered to be LRECL=X and will signal Extended Logical Record Interface processing. On tape output, if OPEN processing has recognized that LRECL=X is in use, OS Simulation will set the tape label LRECL value to 99999.
10. If you specify a value for the BLKSIZE option less than the actual record length, data is truncated to the length you specified as the BLKSIZE, with no warning being given by CMS.
11. 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.
12. Unless the LRECL is specified on the output FILEDEF, the EOV2/EOF2 label contains the record length of the last record written for record format D, DB, V, or VB tape files.
13. The D record format can be used only through BSAM or QSAM and is available only for ANSI tapes.

Usage Notes for Options DISP MOD, OPTCD Q

1. DISP MOD adds records to tape files, for IBM Standard Label tapes only, and to DISK files accessed in WRITE mode. The DISP MOD function has the same effect as an OS 'OPEN EXTEND' macroinstruction.

   ```
   filedef filea 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. If volume switching is necessary to find the real end of file for the file, either DMSTVS for DMSTVI will be called to mount subsequent tape volumes.
2. For DISK files, the file FST information will be used to verify and set RECFM, LRECL, and BLKSIZE options.
3. The RECFM, LRECL, and BLKSIZE defined on the tape HDR2 label will override any values entered on the DCB, because the data being added must match the form of the existing file. If you try to write data that does not match, it may be unreadable when processed for input.
4. 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 the records are added to the correct tape file. If the FID does not match the tape file ID, message DMSTLM434E is issued.
5. DISP MOD is not supported for AL or AUL tapes.
6. When OPTCD Q is used to invoke ASCII/EBCDIC translations, you must not issue multiple BSAM/WRITE instructions for the same record, or translation errors will occur. This is because the first WRITE instruction issued translates the output data into ASCII, in place, within the user data area.
7. The FILEDEF must match the attributes of the file. When using the OPTCD Q option to invoke ASCII/EBCDIC translations, you must ensure the FILEDEF statements are correct. Tape ANSI/ASCII data is assumed to be variable unless otherwise specified by the FILEDEF. If the FILEDEF does not match the attributes of the file, unpredictable results may occur due to the expectation of a variable format file for the translation.

Usage Notes for Options LEAVE, NOEOV, ALT, Recording Format

1. These options control the recording format of tapes you create using OS simulation: 3590C, 3590B, 3490C, 3490B, 18TRACK, XF, DEN 6250, DEN 1600, DEN 800, DEN 38K, 9TRACK, COMP, NOCOMP. For more information on the recording formats, see *z/VM: CMS User's Guide*.

2. In general, you should specify only one recording format option on the FILEDEF command, but for compatibility with previous levels of VM it is allowed to specify 18TRACK together with DEN 38K or 9TRACK together with DEN 800, DEN 1600, or DEN 6250.
3. One thing in particular to watch out for when creating tapes with OS simulation is that open processing may position the tape beyond the beginning of the volume, thus limiting your ability to affect the recording format even with these options. This happens when you use certain tape options such as the ones which allow you to position to a particular file, for example BLP 5.
4. If the device is not capable of writing the recording format you specify, the first I/O operation to it will fail (even if it is not attempting to write on the tape). The FILEDEF command will never fail because of recording format incapability. (This is necessary because the virtual tape device you eventually use need not be attached at the time you issue the FILEDEF command, so CMS has no way to know what it is capable of.)
5. If you do not specify any recording format options on the FILEDEF command, CMS selects a recording format for you. CMS's choice may be a result of a previous TAPE command, so you may use the FILEDEF command with the TAPE command to select tape recording formats for use with OS simulation. For more information on the TAPE command, see *z/VM: CMS User's Guide*.
6. Before you write programs that handle labeled tapes, read the section *Tape Labels in CMS* in *z/VM: CMS Application Development Guide for Assembler*.
7. LEAVE option

   indicates a tape containing 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 AL, AUL, 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: You must position the tape properly yourself before using the LEAVE option. LEAVE may be used with AL, AUL, 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: For the BLP files you can obtain the effect of LEAVE by defining the file as LABOFF rather than BLP.

   Use of the LEAVE option with the DISP MOD option or with the OPEN macro EXTEND option should be avoided. The system needs to read the tape file header labels to verify the new file attributes against the old attributes before appending the new data to the existing file. LEAVE prevents the system from repositioning the tape to read the headers. Depending on the current tape position, this can cause either of the following:

   - An error return from OPEN
   - A separate new file to be written, rather than appending to the old file
8. NOEOV option

   specifies 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 AL, AUL, SL, and SUL files, also writes an EOV1 label and another tape mark after the first tape mark. The tape is then rewound and unloaded. NOEOV suppresses this limited EOV processing. In OS simulation, if you specify the NOEOV option, it is ignored during end-of-volume processing.
9. ALT option

   requests CMS's alternate tape drive function as part of end-of-volume processing (for IBM standard label or ANSI label tape only) and identifies a second tape device to use for it.

Example

```
filedef filea tap1 sl (alt tap2
```

Where:

TAP2
specifies the alternate drive where the second volume of the tape file is mounted.

When an EOV condition occurs, TAP2 will be used. At this point, TAP2 becomes the primary drive and TAP1 becomes the alternate drive. To make TAP1 the primary drive, you must reissue the FILEDEF command.

Usage Notes for the SYSPARM Option

1. For tape, SYSPARM passes the address of the character value *string* to the VSAM emulator or DMSTVI (a system interface routine. For more information about DMSTVI, see *z/VM: CMS Application Development Guide for Assembler*.) For VSAM, the string is passed to the VSAM emulator when the file is opened.
2. 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 65535 bytes long. If it is longer, it is truncated. The string cannot contain blanks or parentheses.
3. 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:
   ```

4. 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.
5. 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. For more information on the CMSCALL macro, see *z/VM: CMS Macros and Functions Reference*.

Usage Notes for the CONCAT Option

1. If you choose to concatenate macro libraries, the first FILEDEF command for concatenated macro libraries should describe the first library in the GLOBAL command.
2. 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.
3. When you concatenate TXTLIBs, all the TXTLIBs listed on the GLOBAL command are searched and the TXTLIB specified on the FILEDEF command is ignored. The GLOBAL command determines the order in which the libraries are searched.
4. You can also concatenate two or more TXTLIBs as the input of the LKED command. The *ddname*, in this case, must be SYSLIB.
5. The global list can contain both CMS disk resident libraries and OS/MVS disk resident libraries. CMS will determine the largest block size library in the global list during OPEN processing. This removes the former restriction which forced the placement of any OS/MVS library to be first in the FILEDEF concatenation and first in the global list.
6. The global list, not the FILEDEF concatenation, determines the library search order. The library must exist in the global list or it will not be searched, even if a FILEDEF concatenation was done to include other library names.
7. For more information on the concatenated macro libraries, see *z/VM: CMS Application Development Guide for Assembler*.

Responses

Entering FILEDEF with no operands or options displays all the file definitions in effect. For example:

```
ddname  device (additional information by device type)
   ⋮       ⋮
```

Note: Although FILEDEF with no operands or options is equivalent to QUERY FILEDEF in most respects, only QUERY FILEDEF can display RECFM, LRECL, and BLKSIZE attribute information. For more information, see "QUERY FILEDEF" on page 684.

Just the *ddname* and *device* are displayed for device types TERMINAL, PRINTER, PUNCH, READER, and DUMMY. Additional information is displayed for the following device types:

```
ddname  DISK fn ft fm [dataset-name]
ddname  TAPn labeltype [n [VOLID volid]|fn]
ddname  GRAF vdev
ddname  VSAM fn ft [emulator-name]
```

If no file definitions are in effect, the following message is displayed at the terminal:

```
DMSFLD324I  No user defined FILEDEFs in effect
```

Following are additional messages you could receive from FILEDEF, and their meanings:

```
DMSFLD069I  Filemode mode not accessed
```

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 dataset-name
```

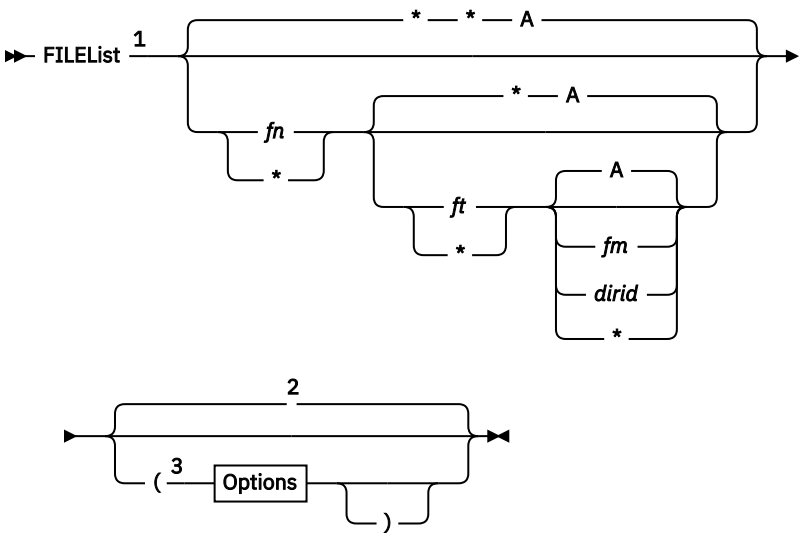
This message is displayed when you issue a FILEDEF command for an OS data set that contains user labels. The message is displayed the first time you issue the FILEDEF command after accessing the disk on which the data set resides.

Messages and Return Codes

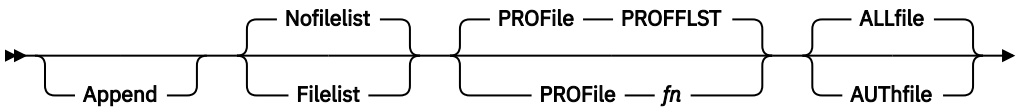
- DMS003E Invalid option: *option* [RC=24]
- DMS023E No filetype specified [RC=24]
- DMS027E Invalid device *devtype* [for SYSaaa] [RC=24]
- DMS029E Invalid parameter *parameter* in the option *option* field [RC=24]
- DMS050E Parameter missing after DDNAME [RC=24]
- DMS065E *option* option specified twice [RC=24]
- DMS066E *option1* and *option2* are conflicting options [RC=24]
- DMS070E Invalid parameter *parameter* [RC=24]
- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS221E Invalid data set name [RC=24]
- DMS224E Fileid already in use [RC=24]

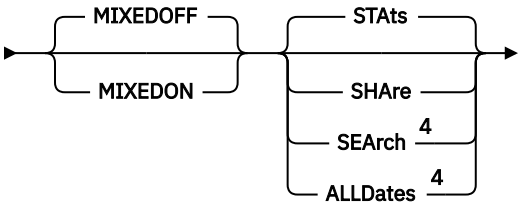
- DMS420E NSL exit filename missing or invalid [RC=24]
- DMS447E Invalid sysparm information [RC=24]
- DMS699E No filetype specified or *vdev* is an invalid disk address [RC=24]
- DMS735E Primary and alternate tape drives are identical. [RC=24]
- DMS1303E Alternate-VSAM emulator *name* is active [RC=24]
- DMS1303E Alternate-VSAM emulator *name* is not available [RC=24]

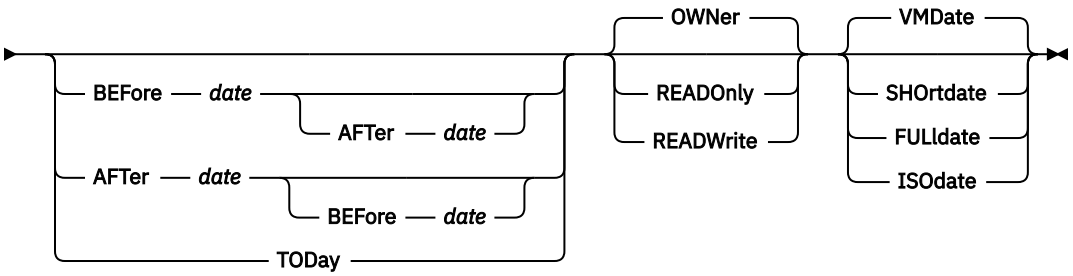
FILELIST



Options







Notes:

[1] Special commands that can be used in the FILELIST environment are EXECUTE, DISCARD, ALIALIST, and AUTHLIST.
[2] The default options you receive appear above the line in the Options group.
[3] You can enter Options in any order between the parentheses.
[4] If this option is chosen, you cannot use the BEFORE, AFTER, TODAY options.

Authorization

General User

Purpose

Use the FILELIST command to display a list of information about CMS files residing on accessed disks and accessed (or directly referenced) 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 XEDIT. You can use XEDIT subcommands to manipulate the list itself. You can also issue CMS commands against the files directly from the displayed list.

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. This is the default.

Certain special characters (* and %) can be used as part of the file name to request the list contain a specific subset of files. For more information on using these characters, see "Using Pattern Matching to Specify Sets of Files" on page 14.

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. This is the default.

Certain special characters can be used as part of the file type to request the list contain a specific subset of files. For more information on using these characters, see "Using Pattern Matching to Specify Sets of Files" on page 14.

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. This is the default. If an asterisk is coded, all accessed disks and directories are searched.

dirid
is the directory identifier of the file(s) for which information is to be collected. For more information on how to specify *dirid*, see "Naming Shared File System (SFS) Directories" on page 6.

If you do not specify any operands, the list contains all files accessed as file mode A in a minidisk or SFS directory. Issuing FILELIST with no operands is like issuing *filelist * * a*.

Options

Append
specifies the list of files should be appended to the existing list. This option is meaningful only when issued from within the FILELIST environment; it cannot be used for a directory that is not accessed. 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 if it is not the default; such as,

```
filelist * * c (share append
```

Filelist
specifies *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 "Using Pattern Matching to Specify Sets of Files" on page 14.

For information on creating and saving a list of files, see Usage Note "7" on page 298.

Nofilelist
specifies *fn ft fm* is not a list of files. This is the default.

PROFile *fn*
specifies the name of an XEDIT macro to be executed when XEDIT is invoked by the FILELIST command. If not specified, one of these macros are used:

- PROFFLST
- PROFFSHA
- PROFFSHR
- PROFFDAT

For more information on which macro is selected under what circumstance, see Usage Note "15" on page 301.

ALLfile
specifies all the files in an SFS directory to be listed, including those you have no authority to. Subdirectories, erased or revoked aliases and external objects are also listed. The default is ALLFILE when the specified file mode refers to a directory, but this option is ignored when the file mode refers to a disk.

AUThfile
specifies only the files in an SFS directory you have read or write authority to are to be listed. AUTHFILE is ignored if the file mode refers to a disk. Subdirectories, erased or revoked aliases and external objects are **not** listed.

MIXEDOFF
specifies the FILELIST environment being entered will not allow any invocation from the command prefix area on mixed case file IDs. No commands will work on these files, and if any are attempted, this message will appear on the line that was typed on:

```
'** Fileid in Mixed Case; invalid for EXECUTE **'
```

This option is ignored if specified with the APPEND option.

MIXEDON
specifies the FILELIST environment being entered will allow invocation from the command prefix area on mixed case file IDs. This allowance, however, is restricted to these four commands:

- ERASE
- RENAME
- COPYFILE
- DISCARD

If anything other than these four commands are attempted, this message will appear on the line that was typed on:

```
'** Invalid command entered on Mixed Case Fileid **'
```

This option is ignored if specified with the APPEND option.

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

For more information on using the STATS option, see "Examples" on page 308. This is the default.

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, or external object)
- 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. For more information on using the SHARE option, see "Examples" on page 308.

SEArch
searches a directory structure for the specified file(s). This option is useful if you know a file name but not the directory 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

For more information on using the SEARCH option, see "Examples" on page 308.

The search begins with the directory you specify as *fm*, and continues within that directory structure for all subdirectories for which you have read, write, DIRREAD, or DIRWRITE 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 even if it is not accessed. You cannot specify an asterisk (*) for the file mode when you use this option.

The SEARCH option cannot be specified with these options:

- BEFORE
- AFTER
- TODAY

ALLDates
specifies this file information:

- File identifier
- Date of creation
- Time of creation
- Date of last reference
- Date of last update
- Time of last update

The ALLDATES option cannot be specified with these options:

- BEFORE
- AFTER
- TODAY

If any of the FILELIST entries are subdirectories, erased aliases or revoked aliases, the date of creation (Create-Dt), time of creation (Create-TM), and date of last reference (Lref-Dt) columns contain dashes. If any of the FILELIST entries are external objects, the date of last reference (Lref-Dt) column contains dashes. If you use the ALLDATES option to list files on a minidisk, the date of creation (Create-Dt), time of creation (Create-TM), and date of last reference (Lref-Dt) columns will contain dashes. For more information on each column displayed, see "Responses" on page 311.

BEFore *date*
lists only those files last written to *before* the date specified. The date can be specified in *mm/dd/yy*, *mm/dd/yyyy*, or *yyyy-mm-dd* format.

Where:

mm
specifies the month

dd
specifies the day of the month

yy
specifies the 2-digit year

yyyy
specifies the 4-digit year

The variables *mm* and *dd* are one or two digit numbers.

The BEFORE option cannot be specified with these options:

- SEARCH
- ALLDATES

AFTer *date*
lists only those files last written to *after* the date specified. The date can be specified in *mm/dd/yy*, *mm/dd/yyyy*, or *yyyy-mm-dd* format.

Where:

mm
specifies the month

dd
specifies the day of the month

yy
specifies the 2-digit year

yyyy
specifies the 4-digit year

The variables *mm* and *dd* are one or two digit numbers.

The AFTER option cannot be specified with these options:

- SEARCH
- ALLDATES

Note: When both the BEFORE and AFTER options are used:

- If the AFTER date precedes the BEFORE date, the output is the set of files last written to between the two dates.
- If the BEFORE date precedes the AFTER date, the output is the set of files last written to either before the BEFORE date or after the AFTER date; the files last written to between the two dates are excluded.
- If the same date is specified for BEFORE and AFTER, the output is the set of files last written to on any date *except* the specified date.

TODay
lists only those files last written to on the present day.

The TODAY option cannot be specified with these options:

- SEARCH
- ALLDATES

OWNer
specifies that if a temporary access of a directory is necessary, it will be done based on your ownership of the directory. This is the default.

READOnly
specifies that if a temporary access of a directory is necessary, it will be accessed in R/O status regardless of your ownership or authority to it. If already accessed, the temporary access is not necessary, so this option will be ignored.

READWrite
specifies that if a temporary access of a directory is necessary, it will be accessed in R/W status regardless of your ownership or authority to it. If the R/W attempt fails a R/O access attempt will be made. If already accessed, the temporary access is not necessary, so this option will be ignored.

VMDate
displays the dates in the format specified by the user's default date format setting. This is the default. For more information on the user's default date format setting, see Usage Note "2" on page 297. This option is ignored if specified with the SEARCH or SHARE option.

SHOrtdate
displays the dates in *mm/dd/yy* format.

Where:

mm
specifies the month

dd
specifies the day of the month

yy
specifies the 2-digit year

This option is ignored if specified with the SEARCH or SHARE option.

FULldate
displays the dates in *mm/dd/yyyy* format.

Where:

mm
specifies the month

dd
specifies the day of the month

yyyy
specifies the 4-digit year

This option is ignored if specified with the SEARCH or SHARE option.

ISOdate
displays the dates in *yyyy-mm-dd* format,

Where:

yyyy
specifies the 4-digit year

mm
specifies the month

dd
specifies the day of the month

This option is ignored if specified with the SEARCH or SHARE option.

Usage Notes

1. Tailoring the FILELIST Command Options

You can use the DEFAULTS command to set up options and 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. For more information, see "DEFAULTS" on page 153.

2. If a date format is not specified on the FILELIST command, the default date format is determined by the setting of the DEFAULTS command. The DEFAULTS command default (the initial setting) is VMDATE, which indicates the user's default date format is to be used. The DEFAULTS command setting can be changed by using DEFAULTS SET.

   The default date format for certain CP and CMS commands can be set on a system-wide basis and also for the individual user. The system-wide default date format is set with the SYSTEM_DATEFORMAT system configuration statement. The user's default date format is set with the DATEFORMAT user directory control statement. The system-wide default and the user's default can also be set with the CP SET DATEFORMAT command. If SET DATEFORMAT is not specified for an individual user and there is no DATEFORMAT statement in the user's directory entry, the default format for that user is the system-wide default. The system-wide and user settings can be queried with the CP QUERY DATEFORMAT command.

   The hierarchy of possible date format settings for the FILELIST command, from highest priority to lowest, is:

   - FILELIST command option
   - DEFAULTS command setting or default
   - User default
   - System-wide default

3. XEDIT Environment

   When you invoke the FILELIST command you are placed in the XEDIT environment, editing a file *userid FILELIST* which resides on your first R/W file mode or on S1 if no R/W mode is accessed.

   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, and so forth.

   However, some XEDIT subcommands are inappropriate in this environment. Subcommands that alter the format or the contents of *userid* FILELIST. For example, SET TRUNC, SET FNAME, SET FTYPE, SET FMODE, or SET LINEND may cause unpredictable results.

4. 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.

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

   ```
   exec filelist
   ```

6. File Sharing Considerations

   While you have your FILELIST displayed on the screen, other users may modify the files (if they are authorized to do so). Remember also, base files may have aliases, and when the base file is updated, the alias is also changed. 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:

   ```
   DMS654E  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.

Note: Always clearing the line following your command will prevent this message from ever occurring.

7. 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 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 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 you saved can be specified in the SENDFILE command issued with the FILELIST option, see "SENDFILE" on page 888. 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 FILELIST option.

8. 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, and typing 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 information in the line. You may type over the entire line displayed, up to column 79. When you are finished typing the command, erase the rest of the line by pressing the ERASE EOF key, or space over the rest of the line. Then press Enter. You may also use the DELETE key to erase the rest of the line, but do not use it to erase only part of the rest of the line. For more information on how the command is interpreted, see "EXECUTE" on page 1393.

   When you press Enter, 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 "Responses" on page 311.

   You may want to enter commands from the FILELIST command line before executing commands 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.

9. 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 Within Other Commands," on page 1385.

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. For more information on using the symbols, see "Examples" on page 308.

    These symbols can be used:

    /
    means one of three things:

    - If a file is displayed on the line, the slash (/) means file name, file type, file mode.

- If a subdirectory is displayed on the line, the slash (/) means + *filemode.filename* for both the STATS and SHARE option screens.
- 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. When in FILELIST, one of the most powerful features of FILELIST can be used:

```
EXECUTE * COPYFILE / = = Z (OLDDATE
```

If you wanted to handle EXEC and XEDIT files, you could do the following:

```
FILELIST * EXEC A             Starts FILELIST
FILELIST * XEDIT A (APPEND            Add all XEDIT files to the
                                      list, issued from FILELIST's
                                      command line
EXECUTE * COPYFILE / = = Z (OLDDATE   Also executed from the
                                      command line
```

You can also exploit the fact that FILELIST is XEDIT based, so before the EXECUTE *, you could change the list of files (for example, remove all files starting with PROF). All commands below are entered in FILELIST's command line.

```
ALL / PROF/       Show all files with filenames beginning with: PROF
DEL *             Remove them from the FILELIST list.
                   (This does not erase the files on the disk)
ALL               Display all lines again
TOP               Go to the top
EXECUTE * ...
```

12. Special Symbols Used Alone

These special symbols can be typed alone on the lines of the FILELIST display. The meanings are:

=
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 Enter).

?
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.)

13. 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,

    ```
    PHILLIPS NOTEBOOK
    PHILLIPSNOTES
    ```

    where PHILLIPS NOTEBOOK is a file and PHILLIPSNOTES is a subdirectory. Issuing,

    ```
    filelist phil* n* a (allfile
    ```

    would find both PHILLIPS NOTEBOOK and PHILLIPSNOTES because PHILLIPSNOTES contains more than eight characters and is matched as if it had a file name of PHILLIPS and a file type of NOTES. Issuing,

    ```
    filelist phillipsnotes * a (allfile
    ```

    would also find both PHILLIPS NOTEBOOK and PHILLIPSNOTES because only the first eight characters are used as the file name. Therefore, any file or alias with the name of PHILLIPS, or any subdirectory with PHILLIPS as the first eight characters in its name would be listed.

    However, if an explicit file name or file type is used, only the file matching it is returned, and no subdirectories. (This is to ensure only one match is found.) For example, issuing:

    ```
    filelist phillips notes (allfile
    ```

    will result in no files being found.

14. The dates and times displayed for an alias are the same as the dates and times displayed for its base file. However, the date of last reference of an alias is not the same as the date of last reference of its base file. The date of last reference of an alias does not get updated when either the alias or its base file is referenced. At the time of creation of an alias, the date of last reference value for that alias takes on the date of last reference value of the base file. This alias date of last reference value does not change. For more information about updates to the date of last reference, see *z/VM: CMS Application Development Guide*.

    Dates and times displayed are local dates and times except for the date of last reference which is in UTC (Coordinated Universal Time).

    When the creation date or time is not a valid date or time value, the conversion to local time does not take place and they are left in UTC.

    The month portion, *mm*, of the dates displayed in both the SHORTDATE format and the FULLDATE format contains a single digit for months 1-9 (January to September). The month portion of the dates displayed in the ISODATE format contains two digits, and for months 1-9 (January to September) the month displayed contains a leading zero.

15. 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: The setting of some keys depends on whether the file mode refers to a disk or directory. Table 15 on page 301 shows the values to which the key are set by PROFFLST XEDIT.

Table 15. Default Key Settings Assigned with PROFFLST XEDIT

Key	Setting	Disk or Directory	Action
Enter	Execute	Both	Execute command(s) typed on file line(s) or on the command line.
PF1	Help	Both	Display FILELIST command description.
PF2	Refresh	Both	Update the list to indicate new files, erased files, and so forth, using the same parameters as those specified when FILELIST was invoked.
PF3	Quit	Disk	Exit from FILELIST.
	Quit	Directory	Exit from FILELIST, or move up one level in the directory structure if you have used PF11 to display the contents of a subdirectory.
PF4	Sort (type)	Disk	Sort by file type, file name.
	Cancel	Directory	Exit from FILELIST regardless of how many levels deep in the directory structure you currently are.
PF5	Sort (date)	Disk	Sort by date and time, newest to oldest.
	Sort (dir)	Directory	Directories are listed by date and time, then files are listed by date and time. The SDIR XEDIT macro does this.
PF6	Sort (size)	Both	Sort by size, largest first.
PF7	Backward	Both	Scroll back one screen.
PF8	Forward	Both	Scroll forward one screen.
PF9	Fl/n	Both	Issue the command FILELIST /n * * at the cursor, so a list is displayed, containing all files that have the file name displayed on the line containing the cursor (all file types and file modes).

Table 15. Default Key Settings Assigned with PROFFLST XEDIT (continued)

Key	Setting	Disk or Directory	Action
PF10	Share/Alldates	Disk Directory	Not assigned. Issue a FILELIST command with either the SHARE or ALLDATES option on the same file(s) specified on the initial FILELIST command. The only time this PF key is assigned to ALLDATES is when the FILELIST STATS environment was entered from the ALLDATES environment through the PF10 key.
PF11	XEDIT/LIST	Both	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.
PF12	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).

Note:

a. If you specify an asterisk for the file mode, or use the FILELIST option,

PF4
is set to Sort (type)

PF5
is set to Sort (date)

PF10
is set to Share

PF11
is set to XEDIT/LIST

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 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.

16. 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: The setting of some keys depends on whether the file mode refers to a disk or directory. Table 16 on page 303 shows the values to which the keys are set by PROFFSHR XEDIT.

Table 16. Default Key Settings Assigned with PROFFSHR XEDIT

Key	Setting	Disk or Directory	Action
Enter	Execute	Both	The same as for the STATS option.
PF1	Help	Both	The same as for the STATS option.
PF2	Refresh	Both	The same as for the STATS option.
PF3	Quit	Disk	The same as for the STATS option.
	Quit	Directory	The same as for the STATS option.
PF4	Sort (type)	Disk	The same as for the STATS option.
	Cancel	Directory	The same as for the STATS option.
PF5	Sort (date)	Disk	The same as for the STATS option.
	Sort (dir)	Directory	The same as for the STATS option.
PF6	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.
PF7	Backward	Both	The same as for the STATS option.
PF8	Forward	Both	The same as for the STATS option.
PF9	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.
PF10		Disk	Not assigned.
	Stats	Directory	Issue a FILELIST command with STATS option on the same file(s) specified on the initial FILELIST command.
PF11	XEDIT/LIST	Disk	The same as for the STATS option.
PF12	Cursor	Both	The same as for the STATS option.

Note:

a. If you specify an asterisk for the file mode, or use the FILELIST option,

PF4
is set to Sort (type)

PF5
is set to Sort (date)

PF6
is set to Auth

PF9
is set to Alias

PF10
is set to Stats

PF11
is set to XEDIT/LIST

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 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.

17. 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. This sets the keys to the values shown in Table 17 on page 304. The SEARCH option is only valid for files in SFS directories.

Table 17. Default Key Settings Assigned with PROFFSEA XEDIT

Key	Setting	Disk or Directory	Action
ENTER	Execute	Directory	The same as for the STATS option.
PF1	Help	Directory	The same as for the STATS option.
PF2	Refresh	Directory	The same as for the STATS option.
PF3	Quit	Directory	The same as for the STATS option.

Table 17. Default Key Settings Assigned with PROFFSEA XEDIT (continued)

Key	Setting	Disk or Directory	Action
PF4	Dirlist	Directory	Issues a DIRLIST /d command for the directory on the line containing the cursor.
PF5	Sort (name)	Directory	Sorts the list of files by file name, file mode.
PF6	Auth	Directory	Issues an AUTHLIST command for the file on the line containing the cursor.
PF7	Backward	Directory	The same as for the STATS option.
PF8	Forward	Directory	The same as for the STATS option.
PF9	Alias	Directory	Issues an ALIALIST command for the file on the line containing the cursor.
PF10	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.
PF11	XEDIT	Directory	Same as for the STATS option.
PF12	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 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.

18. Default Key Settings for ALLDATES Option

Entering the FILELIST command with the ALLDATES option executes the PROFFDAT XEDIT macro, unless you specify a different macro as an option in the FILELIST command. The PROFFDAT XEDIT sets the keys to these values:

Table 18. Default Key Settings Assigned with PROFFDAT XEDIT

Key	Setting	Disk or Directory	Action
ENTER	Execute	Both	Executes the command(s) typed on file line(s) or on the command line.
PF1	Help	Both	Displays the FILELIST command description.

Table 18. Default Key Settings Assigned with PROFFDAT XEDIT (continued)

Key	Setting	Disk or Directory	Action
PF2	Refresh	Both	Updates the list to indicate new files, erased files, and so forth, using the same parameters as those specified when FILELIST was invoked.
PF3	Quit Quit	Disk Directory	Exit from FILELIST. • Exit from FILELIST. • Move up one level in the directory structure if you have used PF11 to display the contents of a subdirectory.
PF4	Sort (type) Cancel	Disk Directory	Sorts the list of files by file type and file name. Exit from FILELIST regardless of how many levels deep in the directory structure you currently are.
PF5	Sort (updt)	Both	Sorts the list of files by date of last update (newest to oldest).
PF6	Sort (size) Sort (lrdt)	Disk Directory	Sorts the list of files by size, largest first. Sorts the list of files by date of last reference (newest to oldest).
PF7	Backward	Both	Scrolls back one screen.
PF8	Forward	Both	Scrolls forward one screen.
PF9	Fl/ n	Disk	Issues the command FILELIST/ n * * at the cursor, so a list is displayed containing all files that have the file name displayed on the line containing the cursor (all file types and file modes).
	S(cdt)	Directory	Sorts the list of files by date of creation (newest to oldest).
PF10	 Stats	Disk Directory	Not assigned. Issues the FILELIST command with STATS option on the same file(s) specified on the initial FILELIST command.
PF11	XEDIT/LIST	Both	Same as for the STATS option.
PF12	Cursor	Both	If the cursor is in the file area, moves it to the command line. If the cursor is on the command line, moves 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 PROFFDAT XEDIT macro sets synonyms 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.

SCRDT
Sorts the list by date of creation and time of creation. Newest to oldest.

SLRDT
Sorts the list by date of last reference, file name, file type, and file mode. Sorts the files newest to oldest, when the dates are the same, sort alphabetically by file name, file type, and file mode.

SUPDT
Sorts the list by date of last update and time of last update (newest to oldest).

SSIZE
Sorts the list by number of blocks and number of records (greatest to least).

19. You can discard files from a file control SFS directory even if that directory is accessed as read-only. To do so, specify the *dirid* on DISCARD. For example, to discard a file from FILELIST on the line where the file is displayed, enter:

```
discard /ntd
```

    Of course, you must have write authority to the directory and the file.

20. If you specify a directory name on the FILELIST command, and the directory is not already accessed, an automatic access will be performed using an available file mode in your search order. When looking for an available file mode, FILELIST will perform the search Z to A. When you exit from the FILELIST screen, the file mode will automatically be released (if FILELIST previously accessed it).
21. The *dirid* operand is supported for all of the FILELIST options, and an automatic access will be performed in all cases if the directory is not already accessed.
22. The FILELIST option uses an input FILELIST file containing file IDs. The *dirid* operand is not supported on these file IDs contained in the file, but is allowed on the file ID used to identify the input file.
23. For the BEFORE and AFTER options, if you are running a release prior to VM/ESA version 2 release 2, and you specify a date of *mm/dd/yy*, the sliding window technique calculates a 4-digit year (*yyyy*) from the 2-digit year, (*yy*) that is input.

    The calculation assumes the 2-digit year is within the 100 year window as calculated by:

```
(current_year - 50) = low end of window
(current_year + 49) = high end of window
```

    For example, if a 2-digit year of 05 is supplied, and the current year (current_year;) is 1997, the window range is between the years 1947 and 2046. In this case, the calculation changes the 2-digit year of 05 to the 4-digit year 2005.

24. To display files using the BEFORE/AFTER/TODAY options, issue:

```
filelist * * a (after mm/dd/yy before mm/dd/yy
filelist * * b (before mm/dd/yy
filelist * * c (today
```

    or

```
filelist * * a (after mm/dd/yyyy before mm/dd/yyyy
filelist * * b (before mm/dd/yyyy
filelist * * c (today
```

    or

```
filelist * * a (after yyyy-mm-dd before yyyy-mm-dd
filelist * * b (before yyyy-mm-dd
filelist * * c (today
```

For example, to display all files between Jan 1, 1991 and Feb 1, 1991, issue:

```
filelist * * a (after 12/31/90 before 2/02/91
```

or

```
filelist * * a (after 12/31/1990 before 2/02/1991
```

or

```
filelist * * a (after 1990-12-31 before 1991-02-02
```

25. The 4-digit years are not supported on a server running a release prior to VM/ESA version 2 release 2. However, for those dates associated with a file on this server, the system sets the year to 19*yy*.

    Where:

    yy
    specifies the 2-digit year
26. An existing FILELIST profile (PROFFLST XEDIT) on your A-disk or in your search order may cause the files in your FILELIST to be sorted incorrectly when sorted by date with FULLDATE or ISODATE in effect. To ensure the date sort function operates correctly, you should erase your old profile and build a new user profile that first calls the system profile for FILELIST (PROFFLST) followed by your customized changes.
27. For more information on how to customize this command, see Appendix A, “Customizing Profiles for CMS Productivity Aids,” on page 1419.

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, ALLDATES and STATS options.

This FILELIST screen was created by issuing the FILELIST command with no operands – which is equivalent to entering:

```
filelist * * a (stats
```

Note: The files are sorted by date and time, newest to oldest, and FULLDATE is in effect.

```
SMITH   FILELIST    A0  V 108  Trunc=108 Size=14  Line=1 Col=1 Alt=0
Directory = SERVER1:SMITH.GOODIES.FOOD
Cmd Filename Filetype Fm Format Lrecl Records Blocks   Date     Time
    PIZZA    TOPPINGS A1 F        107     281     10 10/04/1990 17:59:00
    COOKIE   ASSEMBLE A1 F         98      49      2 10/03/1990 15:17:01
    JELLY    BEANS    A1 F        120     277     10  9/25/1990  9:14:02
    DIETING  TIPS     A1 F         75      28      1  9/24/1990 12:10:03
    CHERRY   PIE      A1 F         80       0      0  9/24/1990  9:09:11
    RECIPES  OLD      A1 -          -       -      -  9/24/1990  9:01:00
    TAILOR   SCRIPT   A1 F        101      50      3  8/14/1990 15:09:05
    CUSTOMER LIST     A1 F         95      34      2  8/04/1990 21:12:04
    SALES             A  DIR       -       -      -  8/04/1990 14:34:34
    SEND     EXEC     A1 F         80     101      4  8/04/1990 15:33:05
    INVENTORIES       A  DIR       -       -      -  8/01/1990 16:50:06
    MYMACRO  XEDIT    A1 V         95      29      2  7/30/1990 20:58:07
    CMSFILES SCRIPT   A1 V         80     489     30  7/26/1990 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 7. Sample FILELIST Screen with STATS Option

For more information on each column displayed, see “Responses” on page 311.

This FILELIST screen was created by issuing:

```
filelist * * a (share
```

```
 SMITH   FILELIST    A0  V 149  Trunc=149 Size=14  Line=1 Col=1 Alt=0
Directory = SERVER1:SMITH.GOODIES.FOOD
Cmd Filename Filetype Fm Owner    Type   R W
    PIZZA    TOPPINGS A1 REYNOLDS ALIAS  X -
    COOKIE   ASSEMBLE A1 PHILLIPS ALIAS  X X
    JELLY    BEANS    A1 STONE    ALIAS  X -
    DIETING  TIPS     A1 SMITH    BASE   X X
    CUSTOMER LIST     A1 HALL     ALIAS  X -
    TAILOR   SCRIPT   A1 PIERRE   ALIAS* X -
    RECIPES  OLD      A1 LEE      EXTRNL - -
    CAKES    INGRED   A1 SMITH    BASE*  X X
    SALES             A  SMITH    DIR    X X
    SEND     EXEC     A1 SMITH    BASE   P P
    INVENTORIES       A  SMITH    DIR    X X
    MYMACRO  XEDIT    A1 SMITH    BASE   X X
    CMSFILES SCRIPT   A1 SMITH    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 8. Sample FILELIST Screen with SHARE Option

For more information on each column listed, see “Responses” on page 311.

This FILELIST screen was created by issuing:

```
filelist * * a (search
```

```
SMITH   FILELIST    A0  V 355  Trunc=355 Size=13  Line=1 Col=1 Alt=0
Cmd Filename Filetype Fm Directory Name
    CMSFILES SCRIPT   A1 SERVER1:SMITH.GOODIES.FOOD
    COOKIE   ASSEMBLE A1 SERVER1:SMITH.GOODIES.FOOD
    CUSTOMER LIST     A1 SERVER1:SMITH.GOODIES.FOOD
    DIETING  TIPS     A1 SERVER1:SMITH.GOODIES.FOOD
    JELLY    BEANS    A1 SERVER1:SMITH.GOODIES.FOOD
    JUNK     FOOD     A1 SERVER1:SMITH.GOODIES.FOOD
    MYMACRO  XEDIT    A1 SERVER1:SMITH.GOODIES.FOOD
    PIZZA    TOPPINGS A1 SERVER1:SMITH.GOODIES.FOOD
    SEND     EXEC     A1 SERVER1:SMITH.GOODIES.FOOD
    STOCK    LIST     -  SERVER1:SMITH.INVENTORIES.
    84       LIST     B1 SERVER1:SMITH.INVENTORIES.PRICES
    85       LIST     B1 SERVER1:SMITH.INVENTORIES.PRICES
    SENDIT   EXEC     -  SERVER1:SMITH.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

====>
                                                        X E D I T  1 File
```

Figure 9. Sample FILELIST Screen with SEARCH Option

For more information on each column displayed, see “Responses” on page 311.

This FILELIST screen was created by issuing:

```
filelist * * a (alldates fulldate
```

(The Filename column is shown with ellipsis (...) in this figure to avoid right-truncation of information.)

```
SMITH   FILELIST    A0  V 400  Trunc=400 Size=4   Line=1 Col=1 Alt=0
Directory = SERVER:SMITH.
Cmd   Fi… Filetype Fm Create-Dt   Create-Tm Lref-Dt    Update-Dt  Update-Tm
      PI… TOPPINGS A1 12/01/1990  12:24:46  10/10/1991 10/04/1991 17:59:0
      CO… ASSEMBLE A1 12/01/1990  12:24:46  10/10/1991 10/03/1991 15:17:0
      JE… BEANS    A1 12/01/1990  12:24:46  10/10/1991 09/25/1991 09:14:0
      DI… TIPS     A1 12/01/1990  12:24:46  10/10/1991 09/24/1991 12:10:0
      MY…          A        -         -         -      09/24/1991 12:10:0



1= Help       2= Refresh  3= Quit    4=Cancel    5= Sort(updt)  6= Sort(lrdt)
7= Backward   8= Forward  9=S(cdt) 10= Stats    11= XED/FILEL  12= Cursor

====>
                                                        X E D I T  1 File
```

Figure 10. Sample FILELIST Screen with ALLDATES Option

For more information on each column displayed, see “Responses” on page 311.

Examples of Using Symbols

These 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 these ways:

- Typed in the *Cmd* area of the screen. The command is executed either by entering EXECUTE on the XEDIT command line and then press Enter, or simply press 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 top a	discard	discard pizza top a
cookie assemble a	assemble /n	assemble cookie

FILE ID	COMMAND	RESULTING COMMAND
jelly beans a	copy / = eggs =	copy jelly beans a jelly eggs a
dieting tips a	copy / /nt b	copy dieting tips a dieting tips b

Responses

Issuing FILELIST with the STATS and ISODATE option displays this information:

```
Filename Filetype Fm Format Lrecl Records Blocks   Date   Time
   fn       ft     fm format lrecl norecs  noblks yyyy-mm-dd hh:mm:ss
   .         .     .  .        .    .       .          .        .
   .         .     .  .        .    .       .          .        .
   .         .     .  .        .    .       .          .        .
```

Where:

fn
specifies the name of the file or directory.

ft
specifies the file type of the file. For a directory this column is blank.

fm
specifies the file mode of the disk or parent directory. A dash indicates the directory is not accessed.

format
specifies the format: Where:

F
specifies the fixed-length

V
specifies the variable-length

DIR
specifies the directory

dash
specifies an erased or revoked alias, or external object.

lrecl
specifies the logical record length of the largest record in the file. For directories, revoked or erased aliases, or external objects, a dash is displayed.

norecs
specifies the number of logical records in the file. For directories, revoked or erased aliases, or external objects, a dash is displayed. For empty files, a zero is displayed.

noblks
specifies the number of CMS data blocks the file occupies. For directories, revoked or erased aliases, or external objects, a dash is displayed. For empty files, a zero is displayed.

Note: The value in *noblks* may not reflect the total amount of space needed to store the file. For SFS files, you can get this information by using the QUERY BLOCKS command.

yyyy-mm-dd
specifies the date (year-month-day) the file was last updated.

Note: The date is displayed in ISODATE format. 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
specifies the time (hours:minutes:seconds) 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. If the type is base file or alias, it is the user ID of the base file owner. For files on disks, this will be the disk owner's user ID. If the type is directory, it is the user ID of the owner of the base file which was erased. If the type is revoked alias, it is the user ID of the owner of the base file which has had all authorizations revoked for the owner of the directory containing the alias. If the type is external object, it is the user ID of the owner of the directory containing the external object.

type
specifies one of these:

MDISK
identifies a file on a disk

BASE
identifies a base file in a directory

DIR
identifies a directory

ALIAS
identifies an alias in a directory

ERASED
identifies an erased alias

REVOKED
identifies a revoked alias

ALIAS*
identifies an alias of a file in DFSMS/VM migrated status

BASE*
identifies a base file in DFSMS/VM migrated status

EXTRNL
identifies an external object.

r
specifies read authority.

X
indicates you have read authority.

–
A dash indicates you do not have read authority.

P
means the authority is managed by an External Security Manager (ESM).

?
means SFS cannot readily determine the authorization.

For files on disk, an *X* will be displayed if the disk is accessed read-only or read/write.

w
specifies write authority.

X
indicates you have write authority.

—
A dash indicates you do not have write authority.

P
means the authority is managed by an External Security Manager (ESM).

?
means SFS cannot readily determine the authorization.

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 you use the SHARE option on a DIRCONTROL directory you have accessed R/O, the authorizations for all the files in that directory will be READ. This is true even if you have directory control write (DIRWRITE) authority to the directory. When you have the directory accessed R/O, you are not able to write to any files. So, read authorization is displayed for every file. To view your true authority, use QUERY AUTHORITY or AUTHLIST commands.

For performance reasons, a question mark (?) is sometimes returned in the *R* and *W* columns. The question mark is returned only for subdirectories when:

- The parent directory has the DIRCONTROL attribute, and
- An External Security Manager (ESM) is not being used, and
- You are not the owner of the subdirectory.

To view your authorities, use the AUTHLIST or QUERY AUTHORITY commands.

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:

```
DMS1182E 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 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 *z/VM: CMS User's Guide*.

If the ALLDATES option is specified, the information displayed is:

```
Filename Filetype Fm Create-Dt Create-Tm Lref-Dt  Update-Dt Update-Tm
fn       ft       fm mm/dd/yy  hh:mm:ss  mm/dd/yy mm/dd/yy  hh:mm:ss
.        .        .  .         .         .        .         .
.        .        .  .         .         .        .         .
.        .        .  .         .         .        .         .
```

Where:

fn
is the name of the file, external object or directory.

ft
is the file type of the file or external object. For a directory this column is blank.

fm
is the file mode of the file, external object or directory.

Create-Dt mm/dd/yy
is the date (month/day/year) the file was created. A dash appears in the column for erased aliases, revoked aliases, or if the files specified are minidisk files rather than SFS files. This column will contain 00/00/00 if the server does not support the Creation date.

Create-Tm hh:mm:ss
is the time (hours:minutes:seconds) the file was created. A dash appears in this column for erased aliases, revoked aliases, or if the files specified are minidisk files rather than SFS files. This column will contain 00:00:00 if the server does not support the Creation time.

Lref-Dt mm/dd/yy
is the date (month/day/year) the file was last referenced. A dash appears in this column for erased aliases, revoked aliases, external objects, subdirectories, or if the files specified are minidisk files rather than SFS files. This column will contain 00/00/00 if the server does not support Date of last Reference.

For more information on updates to the date of last reference, see *z/VM: CMS Application Development Guide*.

Update-Dt mm/dd/yy
is the date (month/day/year) the file was last updated. For a subdirectory, this date is the date the subdirectory was created. A dash appears in this column for erased or revoked aliases.

Update-Tm hh:mm:ss
is the time (hours:minutes:seconds) the file was last updated. For a subdirectory, this time is the time the subdirectory was created. A dash appears in this column for erased or revoked aliases.

These 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  ** Invalid command entered on Mixed Case
                                Fileid **
*   fname   ftype   fmode  has been discarded.
File  fname ftype fmode has been discarded.
*   fname   ftype   fmode  ** Command not issued, FILELIST line
                                has been changed **
```

Messages and Return Codes

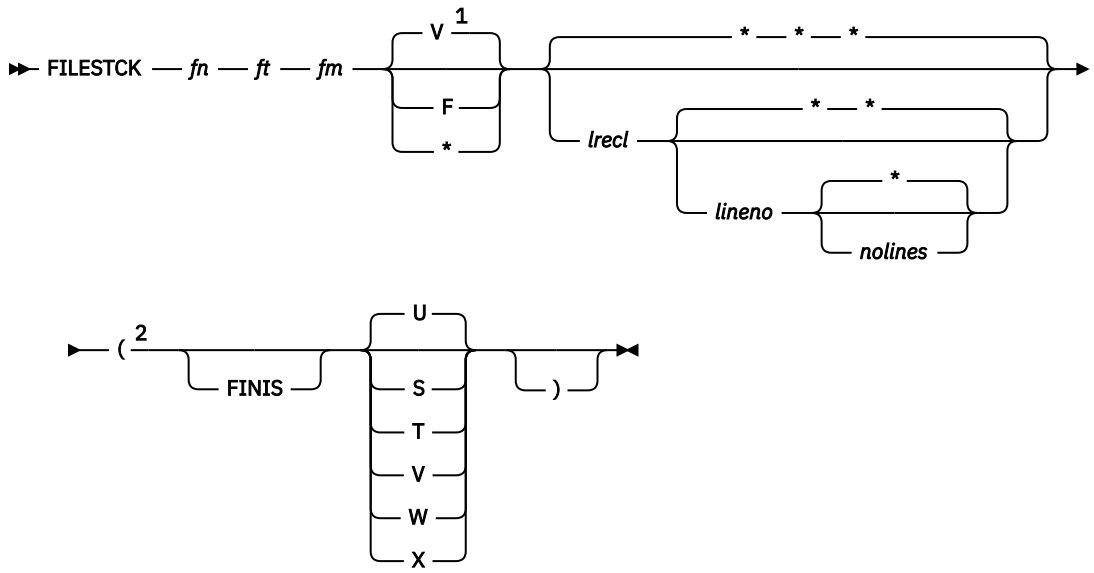
- DMS002E File *fn ft fm* not found [RC=28]
- DMS054E Incomplete fileid specified [RC=24]
- DMS389E Invalid filemode or directory id: *dirid* [RC=24]
- DMS651E APPEND must be issued from FILELIST [RC=40]
- DMS651E APPEND cannot be used with a directory that is not accessed [RC=40]
- DMS653E Error executing LISTFILE, rc=*nn* [RC=*nn*]
- DMS654E Invalid symbol *symbol*; {/0 must be specified alone | invalid character *char* following / symbol [RC=24]
- DMS680E Invalid fileid specified with FILELIST option [RC=20]
- DMS1067E Return code *nn* from the CMS XEDIT command [RC=0]
- DMS1153E File pool *filepoolid* is unavailable or unknown [RC=99]

- DMS1182E The SEARCH option may not be used with a minidisk [RC=74]
- DMS1183E '*' may not be specified for the filemode with the SEARCH option [RC=24]
- DMS1210E Directory *dirname* not found [RC=28]
- DMS1223E There is no default file pool currently defined[RC=40]
- DMS1227E No filemode is available to access directory [RC=0]
- DMS1228E Error executing ACCESS for directory, rc=*nn* [RC=*nn*]
- DMS1229E Directory is empty [RC=0]
- DMS1232E SDIR must be issued from FILELIST Share or Stats screen [RC=40]
- DMS1233E Invalid use of APPEND option [RC=40]
- DMS1234E Error executing FILELIST, rc=*nn* [RC=*nn*]
- DMS1240E You are not authorized to connect to file pool *filepoolid* [RC=76]
- DMS1249I Directory has been temporarily accessed (read/only) as filemode *fm* [RC=0]
- DMS1263E You are not authorized for directory [RC=0]
- DMS2538E File is not in {MACLIB|CSLLIB} format [RC=32]
- DMS3995W You are not authorized to access in read/write mode (RC=4)

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	"Command Syntax Error Messages" on page 1411
Errors in the Shared File System	"File Pool Server Messages" on page 1414

FILESTCK



Notes:

[1] V is the default if not specified or specified as an asterisk.
[2] You can enter Options in any order between the parentheses.

Authorization

General User

Purpose

Use the FILESTCK command to transfer data from the stack to a file. Parameters allow control of format, logical record length, tokenization, and the order in which lines are inserted in the file. Note the parameters given in the following list are positional (parameters skipped over may be given as an asterisk to use default values).

Operands

fn ft fm
specifies the file ID to be created or changed. If the file did not exist before the command execution, a new file is created.

F
V
*****
is the record format for new files (F=fixed, V=variable). If unspecified (or given as an asterisk), new files default to V format. Existing files default to the old format.

lrecl
*****
is the item length for new (or V format) files. If unspecified (or given as an asterisk), the default is the old value for fixed record format (RECFM F), or 130 for variable record format (RECFM V).

lineno
*****
specifies the first (or only) line number of the file to be added or changed. If unspecified (or given as an asterisk), the data is appended to the file.

nolines
*****
is the number of stacked lines to be used. If given as an asterisk, the operation continues until a null line is read. If the value is less than 0, the lines are added in reverse order.

These options specify the LINERD function code to be used to read the stacked line(s).

FINIS
causes the specified file to be closed following completion of FILESTCK.

S
pads the data to 130 characters with blanks (hex 40).

T
reads a logical line (delimited by hex 15 characters).

U
translates to uppercase and tokenizes the data, padding with blanks (hex 40) as necessary. This is the default.

V
translates the data to uppercase.

W
reads a logical line (up to 255 characters).

X
reads a physical line, ignoring line-end characters.

Usage Notes

1. In addition to the FILESTCK command, you can also use the EXECIO command and the preferred CMS Pipelines stack and >> stages to obtain similar results. See "EXECIO" on page 229 and *z/VM: CMS Pipelines User's Guide and Reference* for details.
2. CMS Pipelines offers a more flexible alternative to implement this functionality and more using a combination of built-in programs:

   stack
   Read lines from the input stack.

   >>
   Append to or create a disk file.

   xlate
   Translate the data to uppercase.

   tokenise
   Break up data into individual tokens.

   deblock
   Break lines delimited by line end characters.

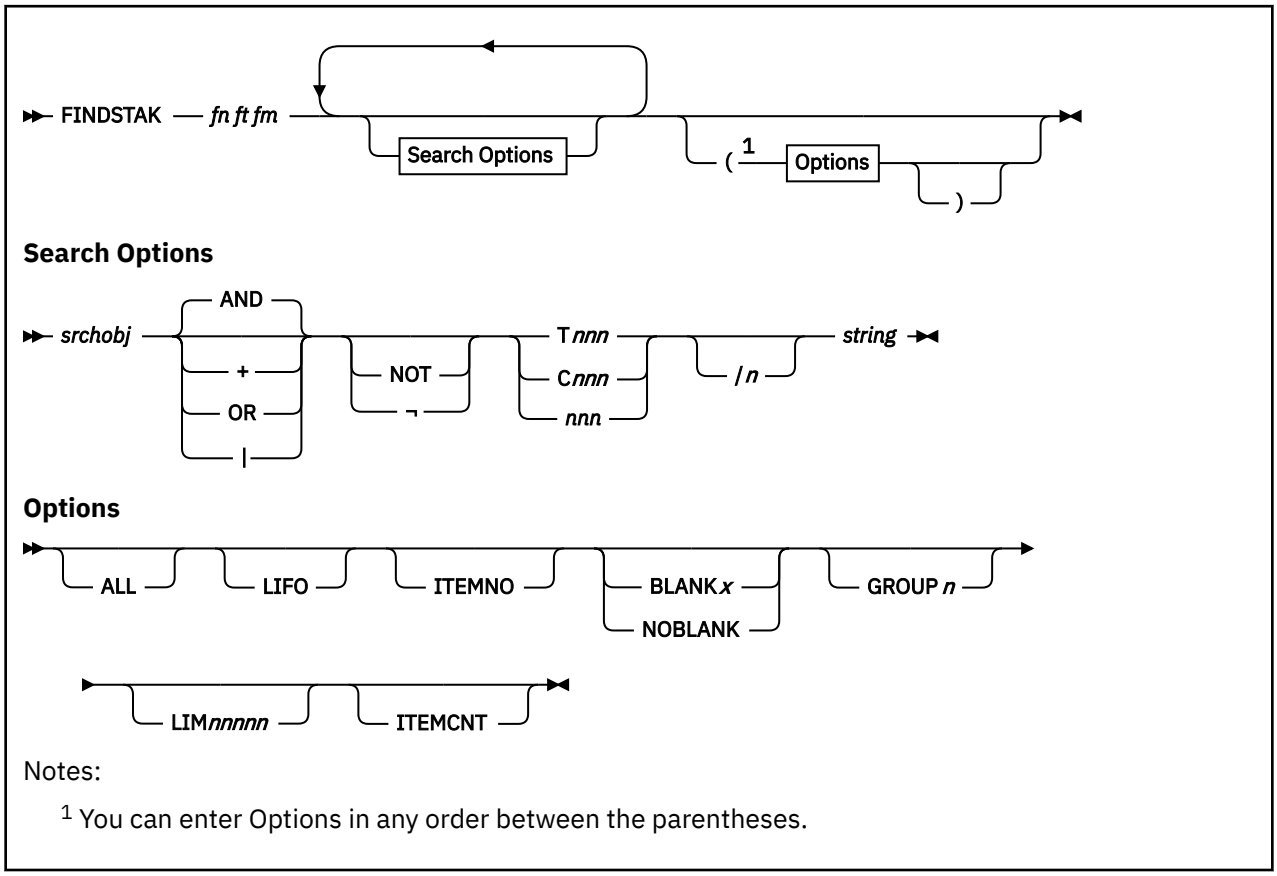
   pad
   Pad the data to a required length.
3. If the FINIS option is not specified, FILESTCK will not close the specified file.
4. FILESTCK loads itself as a nucleus extension.

Messages and Return Codes

- DMS049E Invalid line number *nn* [RC=20]
- DMS109S Insufficient free storage available [RC=4*xxx*]
- DMS389E Invalid *operand: operand* [RC=24]
- DMS389E Invalid integer: *integer* [RC=24]

- DMS394E Invalid option: *option* [RC=24]
- DMS514E Return code *nn* from *command* [RC=1*xxx*]
- DMS618E NUCEXT failed [RC=*rc*]
- DMS2171E Invalid fileid *fn ft fm*. [RC=20]
- DMS2172E Positive line count required (V format files). [RC=20]
- DMS2173E No stacked lines to read. [RC=22]
- DMS2189E DMSRLD failed with return code *rc*. [RC=6*xxx*]

FINDSTAK



Notes:

[1] You can enter Options in any order between the parentheses.

Authorization

General User

Purpose

Use the FINDSTAK command to transfer data from a file to the stack. A large variety of search parameters are allowed to enable record selection based on content. The optional controls are keywords, and may be given in any order; however, they may not be mixed with the search argument values.

Operands

fn ft fm
specifies the file to be searched. This file must be a fixed record length file.

Note: FINDSTAK will give unpredictable results when used on a variable length file.

srchobj
defines what you want to search for. You can specify multiple search objects. The operands and variables you can use to define each object are:

AND
+
OR
|
can be used to specify whether FINDSTAK should look for a record that contains *both* the preceding and the following search strings (AND or +) or a record that contains *either* the preceding or the following search string (OR or |).

The default is AND. If nothing is specified between search parameters, FINDSTAK will look for a record with *both* search strings.

NOT
¬
signals FINDSTAK to look for a record that does not contain the search string in the specified position.

T*nnn*
C*nnn*
nnn
gives the position that FINDSTAK should examine. T*nnn* specifies "token" *nnn* (tokens are words separated by blanks). C*nnn* and *nnn* alone specify column *nnn*. The token or column number *nnn* may be one, two, or three digits long. (You may also see these parameters referred to as "pos1".)

/n
specifies the relative record number. This parameter may only be used if the GROUP*n* option is specified. It tells FINDSTAK to look in each group's *n*th record at the position specified. Otherwise, FINDSTAK will calculate the position starting from each group's first record.

The relative record number, *n*, may be from 1 to 9.

string
is the search string FINDSTAK will look for. Its maximum length is eight characters. (You may also see this parameter referred to as "item1".)

Options

ALL
causes all lines satisfying the search argument(s) to be stacked, rather than just the first line.

LIFO
causes lines to be stacked LIFO (last-in-first-out). The default is FIFO (first-in-first-out).

ITEMNO
stacks the record number of each line following the line itself. If GROUP*n* is specified, the record number of each group's first record is stacked following the group itself.

BLANK *x*
changes the "blank" character to the character given by *x*. This allows a search for an item containing the underscore (_) character, which is the default "blank" character.

NOBLANK
deletes the "blank" character feature altogether (including the default character).

GROUP*n*
allows logical records within a file to occupy more than one physical record. The *n* (a number from 1 to 9) defines the logical record size as *n* physical lines, and the file is then searched *n* records at a time.

LIM*nnnnn*
causes an upper limit (of *nnnnn*) on the total number of lines stacked (excluding the ITEMCNT line, but including lines stacked as a result of the ITEMNO option). The limit may be any whole number from 1 to 99999.

If you specify this option, you do not have to specify the ALL option explicitly.

ITEMCNT
stacks (LIFO) the total number of stacked lines. The total does not include this ITEMCNT line.

Usage Notes

1. Only the first line satisfying the search parameters is stacked unless you specify ALL or LIM*nnnnn*.
2. If you do not specify any search parameters, every record or group of records in the file will satisfy FINDSTAK's selection criteria. The number of lines stacked will depend on the options. Just one line is stacked if no options are given.

3. If FINDSTAK finds a group of records that matches the search criteria, it will stack that group of records.
4. Use the "blank" character (the underscore character, by default) to make sure search strings of fewer than eight characters are matched exactly. For example,

   ```
   findstak fn ft fm t1 abc
   ```

   would match any record beginning with `abc`, including `abcdefg`. To match only records with a first word of `abc`, enter:

   ```
   findstak fn ft fm t1 abc_
   ```

   In this case, only `abc` will match—`abcd` will not.
5. FINDSTAK will resume searching a file at the record following the last record it found, if it is called several times without the ALL option.
6. Whenever FINDSTAK encounters an end of file condition, it will close the current file. This implies that whenever it is invoked with the ALL option, it will close the file (unless some error occurs). Otherwise, the input file will *not* be closed, so a later call referring to the same file may be handled more quickly. If that later call refers to a different file, the former file is closed before the new file is opened.

   If FINDSTAK does not reach the end of file, the file will be *open* for reading when control returns to the calling routine or exec file.
7. Any invalid call to FINDSTAK (return code 20) will cause the next call to start at the beginning of the file.
8. Do not use FINDSTAK to search a file that is open for writing. If you do, FINDSTAK will return the return code it gets from the FSREAD macro.
9. FINDSTAK only looks for uppercase strings. If the file has a string in lowercase, FINDSTAK will not find it.
10. The limit (specified by LIM*nnnnn*) is checked after a record or group of records is stacked, so the number of lines stacked may exceed the limit.
11. If both the BLANK *x* and NOBLANK options are specified, only the one specified last will be in effect; they cannot both be in effect at the same time.
12. In addition to the FINDSTAK command, you can also use the EXECIO command or the preferred CMS Pipelines stack built-in program to obtain similar results. See “EXECIO” on page 229 and *z/VM: CMS Pipelines User's Guide and Reference* for details.
13. FINDSTAK loads itself as a nucleus extension.

Examples

Example 1:

```
findstak test data a 1 n
```

FINDSTAK will stack the first record in the TEST DATA A file that has an `N` in column 1.

Example 2:

```
findstak test data a t2/1 desert + t13/2 orchid + ¬ t15/2 hybrid group2
```

FINDSTAK will search the TEST DATA A file two records at a time for a group of two records that has `desert` as the second token of the first record, `orchid` as the thirteenth token of the second record, and does not have `hybrid` as the fifteenth token of the second record. FINDSTAK will stack the first two-record group that matches the search criteria.

Messages and Return Codes

- DMS109S Insufficient free storage available [RC=4*xxx*]

- DMS514E Return code *rc* from command [RC=1*xxx*]
- DMS618E NUCEXT failed [RC=5*xxx*]
- DMS2189E DMSRLD failed with return code *rc* [RC=6*xxx*]

Return codes:

RC
Meaning

0
Command complete-no errors

1
No record found (end of file)

20
Invalid parameter list

21
File item length > 255–cannot stack records

100
Explanation complete (when '?' specified)

xx
Error xx reading data file (RC from FSSTATE, FSOPEN, or FSREAD)

4xxx
xxx is RC from DMSRLD

4032
Storage not available for header record buffer

4033
Storage not available for module

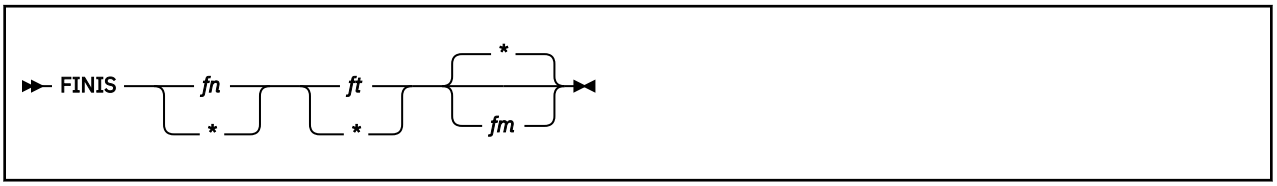
4034
Storage not available for RLD buffer

1xxx
Return code xxx from FSSTATE

5xxx
Return code xxx from NUCEXT

6xxx
Return code xxx from DMSRLD

FINIS



Authorization

General User

Purpose

Use the FINIS command to close one or more files on a disk or in a Shared File System (SFS) directory.

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. You must check the return code to verify file updates have been made, especially when your application is using SFS files. A nonzero return code could mean the work unit has been rolled back and file updates have been lost.
2. 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.
3. When using FINIS to close a file residing in the Shared File System, changes are committed based on the default work unit that was in effect at the time the file was opened. This is a coordinated commit; all changes to protected resources on the work unit are committed in unison (or rolled back, if any of the resources cannot commit). However, the commit does not take effect until the last file open:
   - For output on a work unit is closed
   - On a work unit is closed

   This applies only to files that have been opened through non-SFS statements or macros (for example: CMS FS macros, EXECIO command).
4. The FINIS command closes files opened by the FSOPEN macro, or the EXECIO command within execs. For more information on the FSOPEN macro, see *z/VM: CMS Macros and Functions Reference*. Files opened by the DMSOPEN CSL routine must be closed by the DMSCLOSE CSL routine. Since REXX uses the CSL routines for I/O, FINIS can not be used to close files opened through REXX.

Messages and Return Codes

- DMS1144E Implicit rollback occurred for work unit *workunitid* [RC=31]
- DMS1252T Rollback unsuccessful for file pool *filepoolid*

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in the Shared File System	“File Pool Server Messages” on page 1414

If an error occurs, register 15 contains one of the following error codes:

Code
Meaning

0
One or more files matching the input file ID were closed or one or more files opened using the DMSOPEN or DMSOPDBK interface were committed to disk.

6
No open file(s) were found that match the input file ID, or the file ID (*fn ft fm*) specified was not valid.

31
Close failed for one or more SFS files. Rollback performed on each affected work unit.

FIXCENT

Authorization

General User

Purpose

Use the FIXCENT command to set the century for all the files on a particular minidisk. The century is set according to a sliding window. For more information on the sliding window, see Usage Note "2" on page 325. The FIXCENT command should be used in the cases of:

- Migrations in the Year 2000 or later

  When there are minidisk files that are created or updated in the year 2000 or later, on a VM/ESA release that is not year 2000 ready. After migrating to a VM/ESA year 2000 ready release, these files will appear to have a 4-digit year of 19*xx*.
- Testing of year 2000

  During testing, the minidisk files that are created or updated appear to have a 4-digit year of 20*xx* but actually the 4-digit year should be 19*xx*.

Options

vdev
: is the virtual device number of the minidisk where the files reside. The valid numbers for XA and XC virtual machines are X'0001' through X'FFFF'.

  The default is 191 minidisk accessed at file mode A. If *vdev* is specified, you must specify *fm*

fm
: assigns a one-character file mode letter to all files on the minidisk. The file mode *S* is not valid. The *vdev* is accessed at this file mode. If another disk is accessed at this mode, it is released.

Usage Notes

1. The specified minidisk must be linked R/W.
2. The FIXCENT command will set the century portion of the year according to a sliding window of:

   [*cy-50*,*cy+49*], where integer *cy* is the current year—that is, the year at the moment of the call.

   For example, if a 2-digit year of a file is 05, and the current year (*cy*) is 1997, the window range is [1947,2046]. In this case, FIXCENT changes the 2-digit year of 05 associated with the file to the 4-digit year 2005.
3. If successful, the century of the date of last update of each file on the minidisk is corrected. The changes are saved after all the files have been processed. Either all the dates on the minidisk are processed, or none of them are processed.
4. After this command completes successfully, you should create new data backups because the old ones may not reflect the correct century for the minidisk files.
5. If you are interested in a related command for the file pool server machine, see FILESERV FIXCENT command in *z/VM: CMS File Pool Planning, Administration, and Operation*.

Responses

To set or correct the century for all the files on your a-disk, enter:

```
fixcent
```

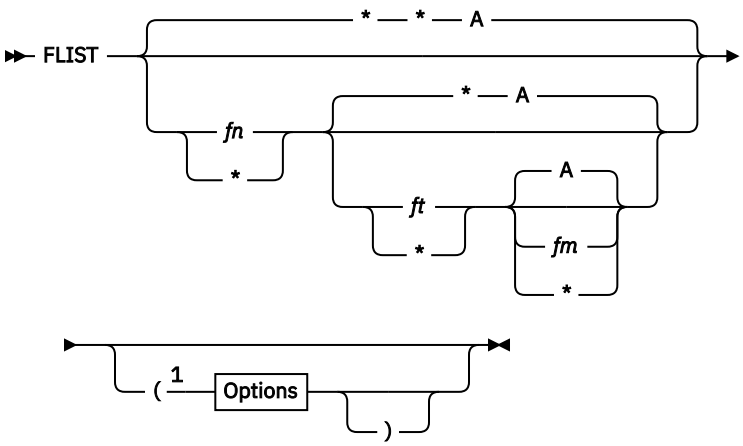
Messages and Return Codes

- DMS037E Filemode *fm* is accessed as read/only [RC=36]
- DMS069E Filemode *fm* not accessed [RC=36]

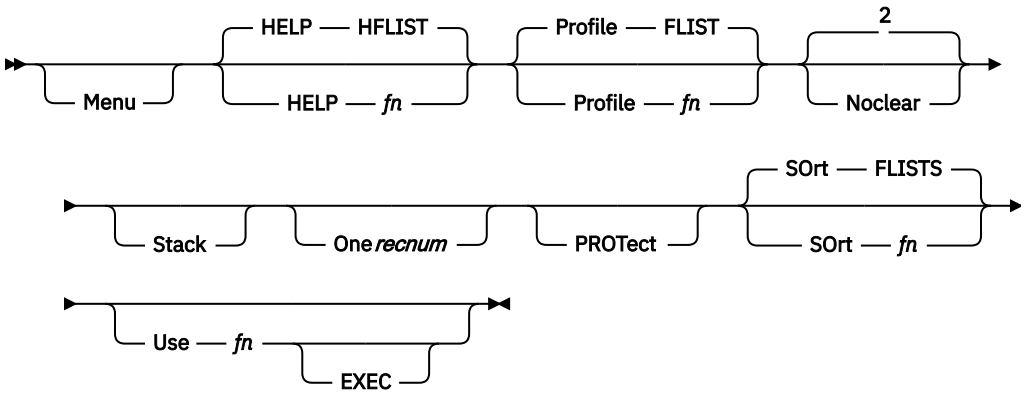
Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

FLIST



Options



Notes:

[1] You can enter Options in any order between the parentheses.
[2] The default is to clear the screen.

Authorization

General User

Purpose

Use the FLIST command to display a full-screen list of information about selected files residing on accessed minidisks and accessed shared file system (SFS) directories. Aliases, external objects, and subdirectories are also listed for directories. Once in FLIST, you can do normal CMS file operations such as EDIT, COPY, and ERASE on the listed files by entering the command in the input area immediately to the right of each file ID on the screen. You can also issue FLIST subcommands to sort the displayed data, or to enter a new FLIST level that displays a full-screen listing of another set of files.

Operands

fn

is the file name, the alias, the external object, or the subdirectory to be listed. The default is an asterisk (*). Specify only the first eight characters of subdirectory names. See "Pattern Matching" on page 329 for information on using * and + to specify a subset of files.

ft
is the file type of the file(s) to be listed. The default is an asterisk (*). Subdirectories will not be listed if a file type other than * is specified.

fm
is the file mode of the file(s) to be listed. The default is A.

Options

Help *fn*
specifies the file name of an exec to be called on invocation of the Help function (/H is a FLIST command). If this option is not specified, the default exec name used is HFLIST.

Menu
is used with the USE option to prevent formatting of the screen to the right of the input areas. When MENU is specified, the text from input file columns 8–27 is placed in screen columns 1–20. The text from file columns 29–68 (for short date) or 29–71 (for full or ISO date) is moved intact to the right of the input areas.

Profile *fn*
specifies the name of an alternate file to be used as the FLIST profile. The file type must be $PROFILE. Anything entered after *fn* is ignored. If *fn* $PROFILE does not exist, or if this option is not specified, the default profile used is FLIST $PROFILE *.

Noclear
specifies not to clear the screen on entry to FLIST. The default is to clear the screen before display of FLIST.

Stack
specifies the following line is to be stacked (and not executed) if the input area is used:

```
*FLIST  nnnnn  filename  filetype  filemode  user_input
```

where:

***FLIST**
will make it a comment if CMS reads it.

nnnnn
is the file entry number in the list.

filename
is the displayed file name.

filetype
is the displayed file type.

filemode
is the displayed file mode.

user_input
is anything the user typed in the input area.

If the input area used is for a subdirectory, the format of the stacked line will be:

```
*FLIST  nnnnn  +fm.ni fm user_input
```

where:

nnnnn
is the file entry number in the list.

+fm.ni
identifies the subdirectory named *ni* in the directory accessed as file mode *fm*.

fm
is the displayed subdirectory's access mode.

user_input
is anything the user typed in the input area.

For more information about the input area, see "Using the Input Areas" on page 333 .

One *recnum*
causes the selected input line to be stacked (implies the STACK option) with any text entered in the input area, then the FLIST session ends. The display starts at the record number specified. If an incorrect value is found for *recnum*, the display starts with the first record. Any extraneous input following *recnum* will be ignored. Following are some occurrences when *recnum* is ignored, and the display starts with the first record:

- No number specified
- Negative numbers or zero are specified as *recnum*
- Number specified as *recnum* exceeds the total number of records
- Any erroneous value or text specified as *recnum*

PROTect
specifies input fields can be used only once.

SOrt *fn*
specifies an alternate exec file name to be called when an FLIST /SORT subcommand (/S*x*) is issued and the USE option is also given. The default file name is FLISTS. A sample FLISTS EXEC is not shipped with FLIST. However, one is shown as a sample under "Using a Sort Exec" on page 336.

Use *fn*
specifies a display of the named file (which must be in CMS EXEC format) rather than FLIST's file listing. The file type must be EXEC and the file mode is assumed to be all (*). Parameters entered after *fn* are ignored.

The data from file columns 8–27 is placed in screen columns 1–20. The remainder of the data columns are placed in the appropriate columns of the display, to the right of the user input areas. This means the LISTFILE DATE or LABEL options should be used when the input file is created. For more information on using the LISTFILE command, see "LISTFILE" on page 447.

Pattern Matching

You can use two special characters when specifying the file name and file type: * and +.

*****
represents any number of characters. As many asterisks as required can appear anywhere in a file name or file type.

For example, if you enter:

```
flist *d* *exec*
```

you will get a list of all your A-disk or directory files whose file name contains a "d" and whose file type contains the character string "exec". The list might include the following files:

```
DAYBREAK EXEC     A
TUESDAY  SAVEEXEC A
FUND     EXECUTE  A
GOODIES  OEXEC2   A
```

You may also substitute an asterisk for file mode (*fn ft* *), to list all files with file name *fn* and file type *ft* on all your accessed minidisks and directories. In addition, you may use an asterisk, followed by a file mode number (*fn ft* *5), to list the files with file name *fn*, file type *ft*, and file mode number 5 on all accessed minidisks and directories. You cannot use an asterisk in place of the file mode number (*fn ft* A* is not valid).

+
means any single character. As many pluses as required can appear anywhere in a file name or file type.

For example, if you enter:

```
flist +++ stock
```

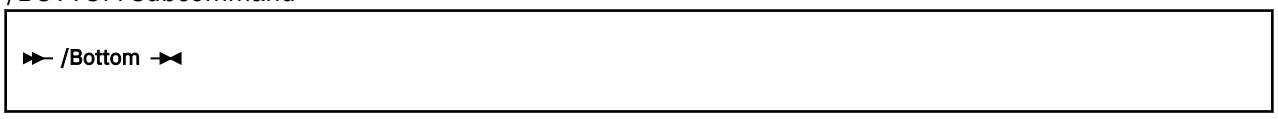
you will get a list of all the files that have a three-letter file name and a file type of "stock" on your A-disk or directory. The list might include the following files:

```
OCT      STOCK    A
FUN      STOCK    A
SUN      STOCK    A
```

FLIST Subcommands

The following subcommands are available to help you once you enter FLIST. (In the syntax diagrams, uppercase letters indicate abbreviations. Only the uppercase letter(s) are needed; all subsequent characters are ignored.)

/BOTTOM Subcommand



The /BOTTOM subcommand displays the last page of the current FLIST level.

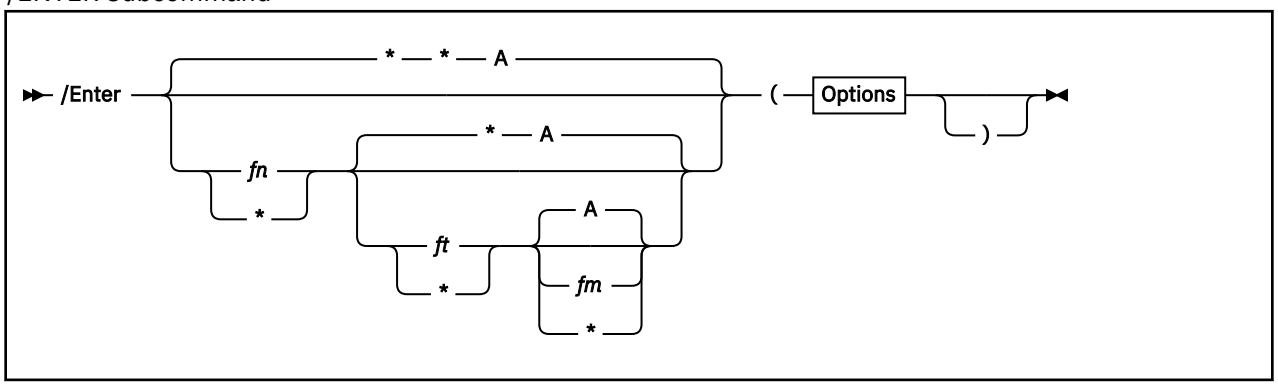
/CANCEL Subcommand

The /CANCEL subcommand cancels all active FLIST levels without processing any other commands entered after the /CANCEL command.

/DSPF Subcommand

The /DSPF subcommand displays the PF key settings for FLIST.

/ENTER Subcommand



The /ENTER subcommand enters another FLIST level.

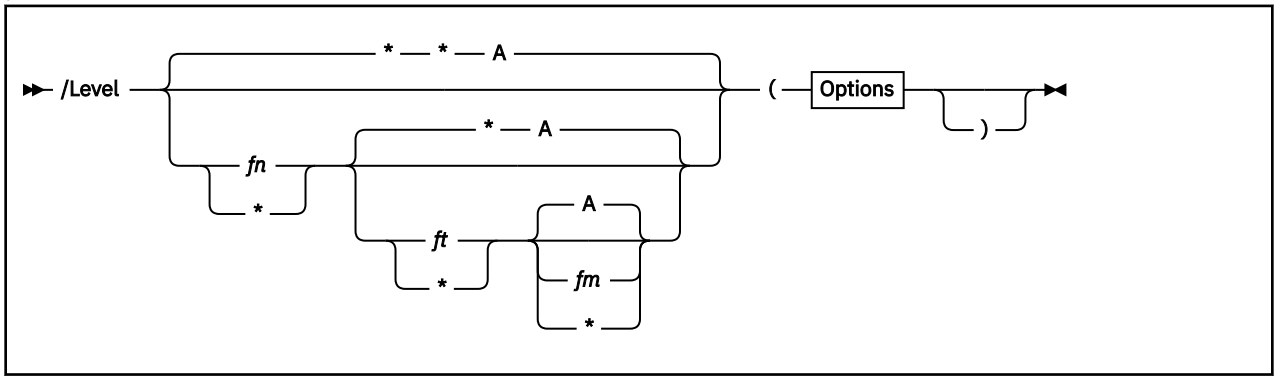
The options are the same as the FLIST options except Profile and Noclear will be ignored if specified. When creating the second level, the screen will be split into two equal parts. To change the location of the split, move the cursor to the place you want the screen to be split and press the SPL PF key (usually PF5).

To display a previous level, use the /E*n* form of the subcommand, where *n* is a digit in the range 0–9 and corresponds to the level desired. The level identifier (LVL *n*) is in the upper left corner of the display.

/HELP Subcommand

The /HELP subcommand displays information on the use of the FLIST command.

/LEVEL Subommand



The /LEVEL subcommand enters another FLIST level.

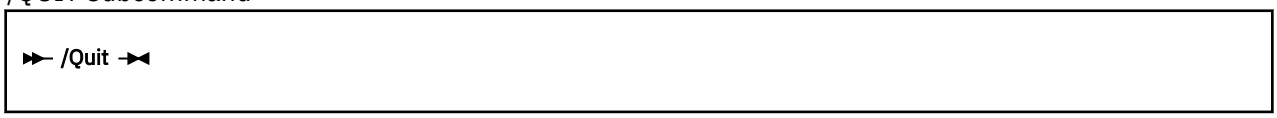
The options are the same as the FLIST options except Profile and Noclear will be ignored if specified. When creating the second level, the screen will be split into two equal parts. To change the location of the split, move the cursor to the place you want the screen to be split and press the SPL PF key (usually PF5).

To display a previous level, use the /L*n* form of the subcommand, where *n* is a digit in the range 0–9 and corresponds to the level desired. The level identifier (LVL *n*) is in the upper left corner of the display.

/OMIT Subcommand

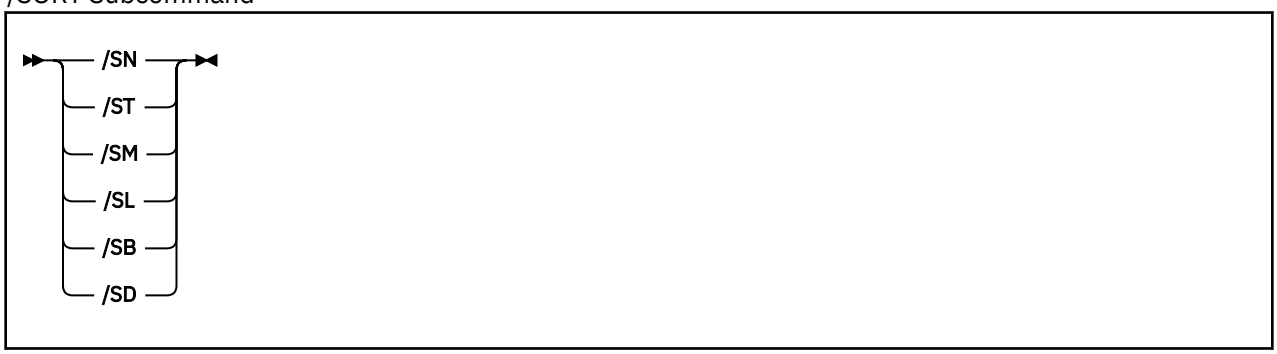
The /OMIT subcommand (as the first token) prevents appending the file ID to the input text and may be used to issue CP or CMS commands. However, specific /n, /t, or /m symbols are substituted.

/QUIT Subcommand



The /QUIT subcommand leaves the current FLIST level after processing entered commands for the level.

/SORT Subcommand



/sn
specifies sorting by file name, file type, file mode.

/st
specifies sorting by file type, file name, file mode.

/sm
specifies sorting by file mode, file name, file type.

/sl
specifies sorting by record length, file name, file type, file mode. (Largest record length first.)

/sb
specifies sorting by number of blocks, file name, file type, file mode. (Largest number of blocks first.)

/sd
specifies sorting by date, time, file name, file type, file mode. (Most recent file first.)

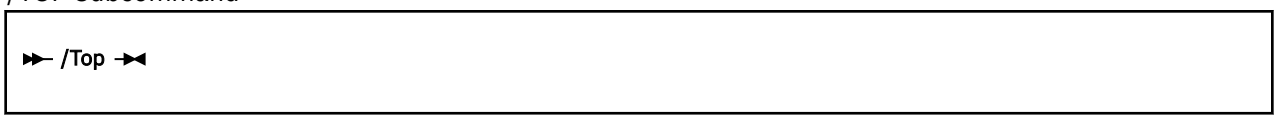
The /SORT subcommand lets you sort the current FLIST level on the specified field(s). After sorting, the level is redisplayed from the top. If entries have been deleted, they are removed during the sort. (See also "Using a Sort Exec" on page 336.)

Attention:

Sorting Subdirectories: Because the file type, record length, and blocks fields are blank for subdirectories, the /st, /sl, and /sb sorts cause subdirectories to appear first, followed by the sorted files. (Blanks come before any other character when sorting.)

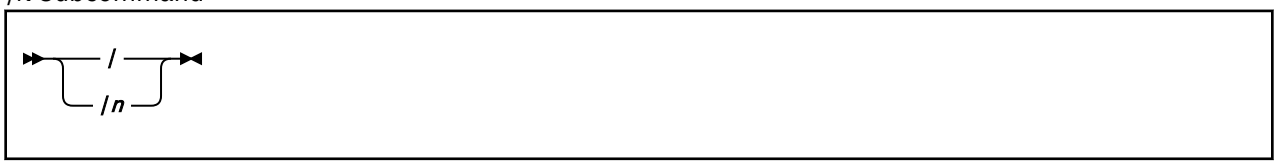
Also, when sorting by file name, only the first eight characters of the subdirectory name are significant. So if you had subdirectories named FAVORITEPIES and FAVORITECAKES, FAVORITEPIES could come before FAVORITECAKES.

/TOP Subcommand



The /TOP subcommand displays the first (top) page for the current level.

/*n* Subcommand



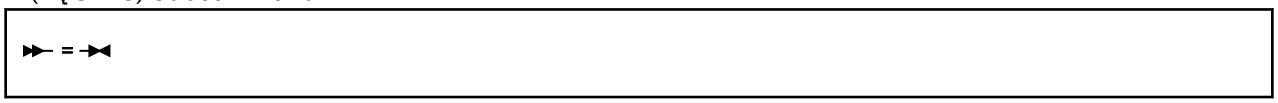
The /*n* subcommand positions the indicated line on the top line of the current level.

where:

/
positions the associated entry on the level top line.

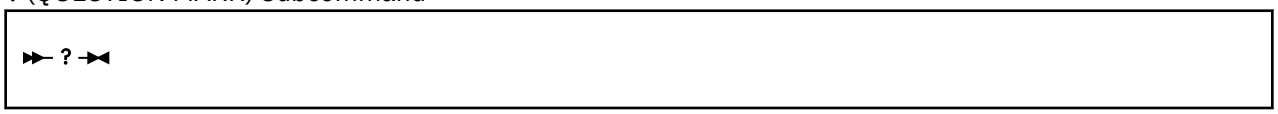
/n
positions the file *n* on the level top line.

= (EQUALS) Subcommand



The = subcommand repeats the last non-FLIST function for the current level. (FLIST functions, save for the /LEVEL, /ENTER, and /OMIT subcommands, cannot be repeated.) The repeat function does not cross level boundaries.

? (QUESTION MARK) Subcommand



The ? subcommand displays the last non-FLIST function for the current level. This allows you to see what command would execute if the = subcommand were entered. To re-execute the displayed command, you must overtype at least one character of the command.

Program Access Keys

You can press the PA keys while you are in FLIST:

PA1
enters CP mode. To return to FLIST, enter:

```
b
```

Note: PA1 will not bring you to CP unless the terminal break key is set to PA1; PA1 is the default setting upon logon. If the terminal break key is not set to PA1, the PA1 key will enter CMS Subset mode just like the PA2 key. For more information about setting the terminal break key, see the CP TERMINAL BRKkey command in *z/VM: CP Commands and Utilities Reference*.

PA2
enters CMS Subset mode. To return to FLIST, enter:

```
return
```

FLIST PF Key Settings

Several PF keys are already set for use with FLIST. The settings and associated FLIST functions follow. Keywords marked with an asterisk are valid **only** when issued from a PF key.

PF Key	Keyword	FLIST Action
1	/H	Obtains information on the use of FLIST.
2	* BRW	Browses the file to the left of the cursor.
3	* END	Ends the level at cursor position.
4	XEDIT	Edits the file at the cursor position with the z/VM System Product Editor (XEDIT).
5	* SPL	Moves the split to the line indicated by the cursor.
6	/SB	Sorts by block size, file name, file type, and file mode.
7	* SCB	Scrolls backward (toward the top of the list).
8	* SCF	Scrolls forward (toward the bottom of the list).
9	/SD	Sorts by date, time, file name, file type, and file mode.
10	/ST	Sorts by file type, file name, and file mode.
11	* >I	Increases input area to end of line.
12	* CAN	Ends all levels and clears the stack if STACK option is in effect.

Using the Input Areas

You may use the input areas after the file IDs to enter any command as if in the normal CMS environment, to execute CMS/CP functions, or to invoke an exec. You do not have to indicate the command is for CP or that it is an exec. However, to prevent any part of the file ID from being used in the command string, use the /OMIT (/O) subcommand as the first parameter.

Note: Commands entered in the input area of an FLIST screen generated with the STACK or ONE option are not executed.

Use a slash (/) if you want all or part of the file ID specified in the user input area. It may be used anywhere in the command sequence, as follows:

/
Insert the complete file ID.

/n
Insert the file name.

/t
Insert the file type.

/m
Insert the file mode.

/f
Insert the subdirectory name (*+fm.ni*).

Any combination of n, t, and m is valid up to a maximum of seven characters.

Note about Subdirectories
Only /m and /f can be used with subdirectories. They can be used in any combination up to a maximum of seven characters. /f is only valid when used next to a subdirectory.

If not explicitly specified, FLIST will append the complete file ID to the user command area. For example, if `erase` alone is entered, FLIST will generate the command string `erase` *`filename filetype filemode`* and pass it to CMS for execution.

Note: You should use the /O subcommand to prevent the file ID from being appended to the input text. If used, /O must be the first token on the line.

After executing a command, the CMS console input stack is processed. Input starting with a slash will replace the original command and will be processed as if it had been entered by the user. Input not starting with a slash will be passed to CMS without any change in the contents of the line. If the stack is empty, the next input field is processed.

When all input fields are processed, the screen is redisplayed and the input area is changed to indicate what happened. If the first character of the input area is displayed as:

*	The command was valid and gave a zero return code.
¬	The specified command gave a nonzero return code. The return code is displayed following the ¬ symbol.
+	An unknown FLIST subcommand was issued.
?	The command was unknown to CP/CMS.

Invalid FLIST functions will remain displayed. If any error condition arises, FLIST will sound the terminal's audible alarm.

Below are examples of using the input area. FLIST uses the date format your virtual machine is using and displays the date in that format (for example: *mm/dd/yy*, *mm/dd/yyyy*, or *yyyy-mm-dd*).

```
APPLE    PIE      A0 erase     V         74         33          1  4/13/00 12:48
RECIPE   EXEC     A1 /tn       V        107         14          1  3/18/00 16:00
VANILLA  ICECREAM A1 copy / /nt b1
SWEET    ROLLS    A2 copy / = = b (rep
COOKIES           A  ac /f o   DIRECTORY                           9/11/99 13:07
INGREDIE ASSEMBLE A5 assemble /n
FUDGE    BROWNIES A2 /o msg Come and get it!
EMPTY    FILE     A1 xedit     V          1          0          0  4/11/99 12:58
DROP     TEST     A1 erase     Dropped or revoked alias
EXTERNAL OBJECT   A1 erase     -          -          -          -  3/09/99  5:42
```

Figure 11. Examples of Using the FLIST Input Area

In the example above, the following commands would be passed on to CMS for execution:

```
erase apple pie a0
exec recipe
copy vanilla icecream a1 vanilla icecream b1
copy sweet rolls a2 = = b (rep
ac +A.cookies o
assemble ingredie
msg Come and get it!
xedit empty file a1
erase drop test a1
erase external object a1
```

Using a FLIST Profile

FLIST looks for a profile (`FLIST $PROFILE *` by default) when it is first invoked. The profile may define PF key functions, the title for level 0, and the text at the bottom of the screen. FLIST will use default values for anything left undefined by the profile.

After processing the profile, FLIST checks the CMS console input stack and treats any stacked records the same way it treats profile records. This means you can use the stack to override the profile definitions.

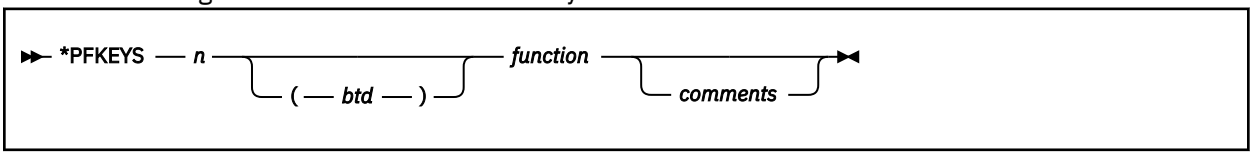
FLIST only processes the profile once. It will ignore the PROFILE option if it is specified with the /ENTER or the /LEVEL subcommands.

Here is what you need to know to create your own FLIST profile:

- The profile is a file of fixed or variable record format with a logical record length (LRECL) of up to 132 characters.
- Use the following record format to define the title for level 0:

  *HEADER *header_text*

  The first 42 characters following *HEADER will replace the level 0 line. The text is centered in the top title (columns 8-49).
- Use the following record format to define PF key functions:



n
specifies the number of the PF key to be set.

(*btd*)
if specified, FLIST will display these three characters on the bottom line of the screen to show this PF key's function. If not specified, FLIST will display the first three characters of the function. However, if the bottom line of the screen is defined—either by the last line in the profile or by the last stacked line—it will be displayed as defined, regardless of the *PFKEYS records.

The (*btd*) format is important. It must be exactly three characters long and placed between parentheses.

function
specifies the FLIST keyword, CP or CMS command, or other command to be assigned to the PF key. Only one PF key or command can be set per record.

Most functions are processed as if they were entered in the input area—except for the following FLIST keywords:

Function	Meaning
SPL	Moves the split to the line indicated by the cursor.

Function	Meaning
BRW	Invokes the BROWSE function.
HLP	Invokes the HELP function (HFLIST EXEC).
SCF	Scrolls forward (toward the end of the file).
SCB	Scrolls backward (toward the top of the file).
>I	Increases user input area to end of line.
CAN	Cancels all levels.
END	Terminates the level containing the cursor.
%%	Clears the definition for the PF key.

These keywords are *only* valid when issued from a PF key. They cannot be abbreviated and must be in uppercase.

comments
explain what the function is. Comments may easily be specified with FLIST keywords. However, because other commands are processed as though they were entered in the input area, any comments will be sent as input to the command and may cause errors. In most cases, putting the comments after a closing parenthesis ()) will solve this problem. For example,

```
*PFKEYS 6 (PRT) PRINT / (NOCC) Print an unformatted file.
```

- To define the bottom line of the FLIST screen, enter what you want displayed as the last record of the profile. The record cannot begin with an asterisk and cannot be more than 132 characters long.

The following is an example of an FLIST profile:

```
*************************************************************************
**                                                                     **
**                        FLIST Profile                                **
**                                                                     **
*************************************************************************
*PFKEYS 01 (HLP) /H   Invoke HELP function
*PFKEYS 02 BRW  BROWSE the file on cursor line
*PFKEYS 03 END  Terminate this level after processing input
*PFKEYS 04 (XED) XEDIT
*PFKEYS 05 SPL  Split the screen (only in multiple levels)
*PFKEYS 06 /SB  Sort by block size (descending)
*PFKEYS 07 SCB  Scroll backward (to top of list)
*PFKEYS 08 SCF  Scroll forward (to end of list)
*PFKEYS 09 /SD  Sort by date (descending)
*PFKEYS 10 /ST  Sort alphabetically by file type
*PFKEYS 11 >I   Increase input area length
*PFKEYS 12 CAN  Cancel all levels, do not process input
*PFKEYS 13 (HLP) /H   Invoke HELP function
*PFKEYS 14 BRW  BROWSE the file on cursor line
*PFKEYS 15 END  Terminate this level after processing input
*PFKEYS 16 (XED) XEDIT
*PFKEYS 17 SPL  Split the screen (only in multiple levels)
*PFKEYS 18 /SB  Sort by block size (descending)
*PFKEYS 19 SCB  Scroll backward (to top of list)
*PFKEYS 20 SCF  Scroll forward (to end of list)
*PFKEYS 21 /SD  Sort by date (descending)
*PFKEYS 22 /ST  Sort alphabetically by file type
*PFKEYS 23 >I   Increase input area length
*PFKEYS 24 CAN  Cancel all levels, do not process input
PF:1 HLP 2 BRW 3 END 4 XED 5 SPL 6 /SB 7 SCB 8 SCF 9 /SD 10 /ST 11 >I 12 CAN
```

Figure 12. Sample FLIST Profile

Using a Sort Exec

If the SORT and USE options are specified when the FLIST command is issued, users should invoke their own sort routines while within the FLIST environment. The exec specified on the SORT option will be given control when the user issues a sort subcommand (/Sn) from the FLIST screen. FLIST will pass four parameters to the user's sort exec. They are:

1. The file name
2. The file type
3. An asterisk (*) as the file mode of the file specified with the USE option of the FLIST command
4. The second character of the sort subcommand issued from the FLIST screen (for example, the "n" in /Sn)

When the exec completes, FLIST redisplays the file specified on the USE option of the FLIST command.

Note: When the USE option is specified creating a sort exec, the CMS EXEC created by the LISTFILE option may be in any of the 3 date formats (*mm/dd/yy*, *mm/dd/yyyy*, or *yyyy-mm-dd*). This should be considered when writing a sort exec.

The following is a sample sort exec you can use with FLIST:

```
/* Sample SORT routine */
Trace Off
/*                                                          */
/* Routine to parse the input parameters, determine the type */
/* of SORT to be done, and perform the SORT of the file.     */
/* This routine assumes the file to be sorted is in CMS EXEC */
/* format. The sorted file retains the input fileid.         */
/*                                                          */
Parse upper arg fn ft fm type .
'SET CMSTYPE HT'
'TYPE' fn ft fm
A = POS(type,'XY')         /* X and Y are defined SORT types */
say A
say type
If A ¬= 0 then Do
   If A = 1 then Queue '8 15'  /* X - SORT file by filename  */
   Else Queue '17 24'          /* Y - SORT file by filetype  */
   'SORT' fn ft fm 'HOLD FILE A'
   'COPY HOLD FILE A' fn ft '= (REP'
   'ERASE HOLD FILE A'
End
Else rc = 111
Exit rc
/* End of EXEC */
```

Figure 13. Sample FLISTS EXEC

The following steps show how you can try out the sample FLISTS EXEC.

1. Be sure you are running in the CMS environment with a R/W minidisk or directory accessed as A.
2. Create a CMS EXEC A file, which you will specify with the USE option of FLIST. (Be sure files exist on the minidisk or directory accessed as A.) Issue the following CMS command:

   ```
   listfile * * a (exec label
   ```

3. Make sure the sample FLISTS EXEC exists on an accessed minidisk or directory.

   Unless someone already created it, you will have to create the FLISTS EXEC.

4. Make sure you do not have a HOLD FILE A file.

   If the file exists, you should rename it before invoking the FLISTS EXEC, because the exec erases HOLD FILE A.

5. Issue the following command to invoke FLIST:

   ```
   flist (use cms sort flists m
   ```

6. From the FLIST screen, issue the sort subcommand /sx. This causes FLISTS EXEC to get control and to sort CMS EXEC by file name. When the sort exec finishes, FLIST redisplays the sorted CMS EXEC.
7. From the FLIST screen, issue the sort subcommand /sy. This causes FLISTS EXEC to get control and to sort CMS EXEC by file type. Again, when the sort exec finishes, FLIST redisplays the sorted CMS EXEC.
8. When you are ready to quit, press PF3 (End) to leave FLIST. If you renamed HOLD FILE A in step 4, you may want to change its file ID back to HOLD FILE A.

Usage Notes

1. Never reaccess a minidisk or directory displayed on one of the levels.
2. Do not erase files displayed on any level with an exec or with an `erase * filetype filemode` sequence.
3. Do not erase a file displayed on more than one level.

   The situations described above cannot be detected by FLIST and may result in randomly displayed file IDs or FLIST termination.
4. Executing commands which alter files other than the subject file may result in randomly displayed file IDs upon return to FLIST. Processing of these commands is not affected.
5. FLIST will not clear the user-supplied I/O interrupt table. Therefore, you can handle your own interrupts if you want to.
6. FLIST loads itself as a nucleus extension.
7. FLIST supports the following screen sizes: 24 x 80, 32 x 80, 43 x 80. Screen sizes other than these are forced to 24 x 80. For example, screen size 27 x 132 is forced to a 24 x 80 screen size.
8. FLIST uses the date format your virtual machine is using and displays the date in that format.
9. In addition to the FLIST command, you can also use the preferred FILELIST command to obtain similar results. See "FILELIST" on page 291 for details.

Examples

1. The following output is an example of using the FLIST command:

```
 LVL 0 - A 191       18000 BLKS 3390 R/W  69%     FILE          1 OF    12
DTRYLST  TXTAQ0   A1           F         80         44          1  12/20/01 10:41
DTRYLST  AUXAQO   A1           F         80          1          1  12/26/01  1:45
DTRYLST  LISTING  A1           F        121       1301          1  12/26/01  1:45
DTRYLST  SJ871AQ0 A1           F         80          2          1  12/26/01  1:44
DTRYLST  TXTAQ0   A1           F         80        445          1  12/26/01  1:45
DTRYSAD  AUXAQ0   A1           F         80          1          1  12/10/01  4:08
DTRYSAD  SJ966QQ0 A1           F         80          1          1  12/18/01 23:42
DTRYSAD  TXTAQ0   A1           F         80         51          1  12/18/01 23:43
FLIST    MAPAQ0   A5           F        100          2          1  12/26/01  1:46
FLIST    MODULE   A2           F       1312          3          1  12/26/01  1:46
FLIST1   $PROFILE A2           V         78         31          1  12/26/01  2:06
GOPAL    NETLOG   A0           V        108         37          1  12/26/01 17:02

 1       2   3          4        5   6   7     8    9     10

 PF: 1 HLP 2 BRW 3 END 4 XED 5 SPL 6 /SB 7 SCB 8 SCF 9 /SD 10 /ST 11 >I  12 CAN
```

Figure 14. Sample FLIST Screen

Area
Description

1
FN – File name.

2
FT – File type.

3
FM – File mode.

4
Area for entering subcommands.

5
FF – File format. Designates how records are arranged in a file. F=Fixed and V=Variable.

6
LRECL – Logical record length.

7
Number of records in a file.

8
Block size.

9
Creation or last modified date.

10
Creation or last modified time.

2. In this example, a CMS EXEC was created using LISTFILE with the DATE option. Then, the FLIST command was issued with the USE option.

```
l * exec (date e
flist (use
```

Following is the sample output. Note the output appears in CMS LISTFILE format:

```
LVL 0 ---------------- CMS      EXEC   *---------- FILE          1 OF     12
$IDMON$  EXEC     A1           V         71          18          1  7/31/01 22:02
$PLIOPT  EXEC     A2           V         68          15          1  5/11/01 14:05
CHCKQREF EXEC     A1           V         35           9          1  7/15/99 21:53
CHANGE   EXEC     A1           V         52         131          2  5/09/00 13:37
CMS      EXEC     A1           V         99          93          2  8/12/00 10:10
DL       EXEC     A1           V         67          18          1  1/20/02 16:24
DMSA4220 EXEC     A1           F         80         110          3  8/27/01 10:36
DZAXLOCI EXEC     A1           V        210         146          2 11/04/01 20:41
GETHELP  EXEC     A1           V         67           9          1  6/03/00 11:51
GETVDISK EXEC     A1           V         28          10          1  3/17/01  7:10
PRINT    EXEC     A1           V         53          12          1  1/30/02 15:08
3820P    EXEC     A1           V         73          13          1  5/23/01 15:08




PF: 1 HLP 2 BRW 3 END 4 XED 5 SPL 6 /SB 7 SCB 8 SCF 9 /SD 10 /ST 11 >I  12 CAN
```

Figure 15. Sample FLIST Screen with USE Option

3. In the following example, a CMS EXEC was created using LISTFILE with the DATE option. The FLIST command was issued using the MENU and USE options.

```
l * exec (date e
flist (menu use
```

Following is the sample output, which also appears in CMS LISTFILE format:

```
LVL 0 --------------- CMS      EXEC     *-------- FILE          1 OF     12
$IDMON$  EXEC     A1           V         71         18          1
$PLIOPT  EXEC     A2           V         68         15          1
CHCKQREF EXEC     A1           V         35          9          1
CHANGE   EXEC     A1           V         52        131          2
CMS      EXEC     A1           V         99         93          2
DL       EXEC     A1           V         67         18          1
DMSA4220 EXEC     A1           F         80        110          3
DZAXLOCI EXEC     A1           V        210        146          2
GETHELP  EXEC     A1           V         67          9          1
GETVDISK EXEC     A1           V         28         10          1
PRINT    EXEC     A1           V         53         12          1
3820P    EXEC     A1           V         73         13          1




PF: 1 HLP 2 BRW 3 END 4 XED 5 SPL 6 /SB 7 SCB 8 SCF 9 /SD 10 /ST 11 >I  12 CAN
```

Figure 16. Sample FLIST Screen with MENU and USE Options

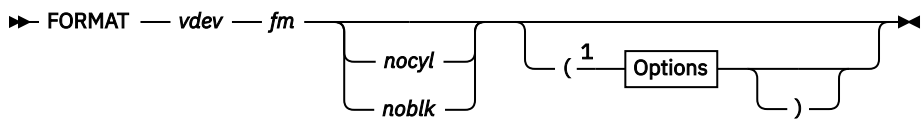
Messages and Return Codes

- DMS002E File not found [RC=28]
- DMS048E Invalid filemode *fm* [RC=24]
- DMS069E Filemode *mode* not accessed [RC=36]
- DMS104S Error *rc* reading file from disk or directory [RC=2*xxx*]
- DMS109S Insufficient free storage available [RC=2|4*xxx*]
- DMS514E Return code *nn* from *command* [RC=1*xxx*]
- DMS618E NUCEXT failed [RC=5*xxx*]
- DMS1153E File pool *filepoolid* is unavailable or unknown [RC=99]
- DMS2189E DMSRLD failed with return code *rc*. [RC=6*xxx*]
- DMS2190E Invalid console type or console disconnected. [RC=1]

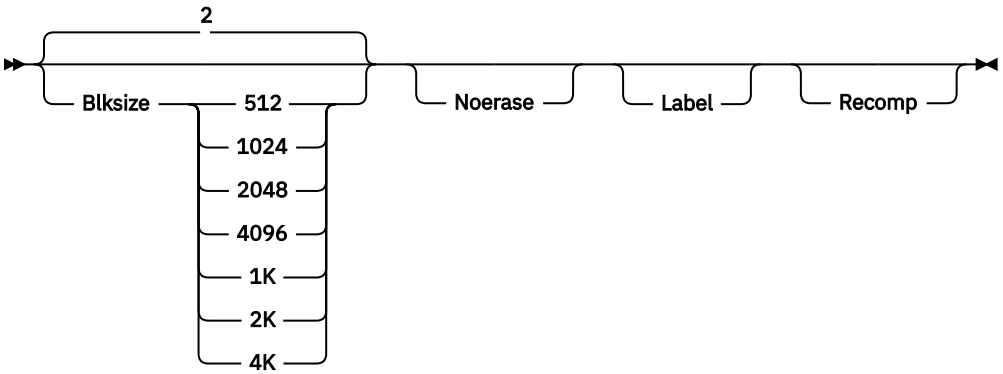
Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

FORMAT



Options



Notes:

[1] You can enter Options in any order between the parentheses.
[2] For more information, see the default for each DASD.

Purpose

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

Authorization

General User

Operands

vdev
is the virtual device number of the minidisk to be formatted. Valid addresses are 0001 through FFFF.

fm
is the file mode letter to be assigned to the specified device address. Any single character letter, A through Z, excluding S, is valid. This field must be specified. If any other disk is accessed at this mode, it is released.

Note: If you specify S as the file mode, you will receive an error, DMSFOR048E Invalid Filemode S.

nocyl
is the number of cylinders to be made available for use. If this operand is omitted, or if the number specified exceeds the actual number of cylinders on the disk, all the cylinders on the disk are made available for use.

noblk
is the number of FB-512 blocks to be made available for use. If this operand is omitted, or if the number specified exceeds the actual number of blocks on the disk, all the blocks on the disk are made available for use. If necessary, the number used will be adjusted downward to the previous multiple of 2, 4, or 8 for a 1 KB, 2 KB, or 4 KB blocksize, respectively, as the number of FB-512 blocks formatted must result in an integral number of CMS blocks.

Options

Blksize
specifies the physical DASD block size of the CMS minidisk. Acceptable values are 512, 1024, 2048, and 4096. The block sizes 1024, 2048, and 4096 may alternately be specified as 1K, 2K, and 4K, respectively.

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 Block size

3380
4096

3390
4096

FB-512 devices, such as 9336, default to a block size of 4096. For more information on choosing an appropriate block size, see Usage Note “6” on page 343.

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 block 1 of an FB-512 device. A prompting message requests a six-character disk label. A valid label is composed of one to six numeric (0-9) or alphabetic (A-Z) characters, or both. A label with fewer than six characters is padded with blanks on the right. If the label specified has more than six characters, only the first six characters are used.

Note: Special characters, such as "(" or (a blank) are valid in a label with FORMAT, but may not be valid in other VM commands.

LABEL can only be used with CMS formatted virtual disks that are accessed. If you choose to change the label (volume serial number) of an OS or DOS (for example, MVS™ or VSE) formatted disk, you must use some other utility to do so, such as Device Support Facilities.

Recomp
changes the number of cylinders/blocks (FB-512 blocks) on the disk that are available to you. If you specify *nocyl/noblk* and that many cylinders/blocks are not available, you only get the available number of cylinders/blocks. If you do not specify *nocyl/noblk*, the maximum number of cylinders or FB-512 blocks **last** formatted on the disk is made available to you.

RECOMP can only be used with CMS formatted virtual disks that are accessed.

Usage Notes®

1. You can use the FORMAT command with any virtual 9336, 3380, or 3390 device.
2. 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.

3. 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.
4. 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.
5. 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 minimum number of FB-512 blocks which must be formatted for minidisks of different sizes are:

   Minidisk Blocksize
   Minimum Blocks Needed

   512
   7

   1024
   14

   2048
   28

   4096
   56

6. When choosing an appropriate BLKSIZE to format a disk, there are two main concerns: space utilization and performance.

   On CKD and ECKD devices, you can allocate more bytes of user data per track using larger block sizes, provided your average file size is larger than the block size. For example, a single cylinder of a 3390 formatted at 4 KB BLKSIZE can hold 180 CMS 4 KB blocks, or 720 kilobytes on a single cylinder. However, when formatted at 512 BLKSIZE, a single cylinder of a 3390 will hold only 735 CMS 512 byte blocks, or 367.5 kilobytes. Thus, larger block sizes are generally preferable for CKD and ECKD DASD for optimum space utilization. However, if you have many small files, a smaller block size might be preferable, because each CMS block can only be used by a single file.

   For FB-512 devices, smaller block sizes will tend to make better use of space on the disk. You get the same number of bytes of space for a given number of FB-512 blocks, regardless of the block size you choose. For larger block sizes, you will generally waste more space for each file, because on average, half of the last CMS block allocated to the file is unused.

   FB-512 devices under FCP SCSI control will have better I/O performance when blocked at larger block sizes. BLKSIZE 4096 is recommended for FCP SCSI.

   A BLKSIZE of 4 KB will optimize the I/O if the disk is to contain large files with no missing records (dense). A BLKSIZE of 1 KB 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 4 KB disk than on a 1 KB disk, therefore, the smaller BLKSIZE is preferable.

7. If you have a minidisk defined as virtual device number 192, it may be automatically formatted and accessed when you IPL CMS. If 192 is:

- An unformatted temporary minidisk or virtual disk in storage, CMS formats it and accesses it as file mode D.
- A CP-formatted temporary minidisk or virtual disk in storage, CMS reformats it for CMS use and accesses it as file mode D.
- A CMS-formatted temporary minidisk, virtual disk in storage, or permanent minidisk accessed as a file mode other than D, CMS reaccesses it as file mode D.
- An unformatted or CP-formatted permanent minidisk, CMS does not automatically format, reformat, access, or reaccess it.

When CMS accesses a 192 minidisk as file mode D, any minidisk or SFS directory already accessed as D is released.

8. To format a DASD for correct Linux® usage, you must use either the CP CPFMTXA utility or the Device Support Facility. For more information, see *z/VM: CP Commands and Utilities Reference*.
9. Because the CMS file system requires file status and control information to reside below 16 MB in virtual storage, there is a practical limitation on the size of CMS minidisks. As a minidisk increases in size, or more files reside on the disk, the amount of virtual storage associated with the disk for CMS file system status and control increases in storage below 16 MB. The current ECKD DASD limitation is 65520 cylinders for a 3390 disk on an IBM TotalStorage™ DASD subsystem, or about 45 GB of data. The maximum size for FBA SCSI disks supported for use by CMS or GCS is 381 GB. IBM suggests that customers defining disks for use by CMS should set a practical limit of about 22 GB. If larger disks are defined, they should be limited to contain very few files, or the CMS file system may not be able to obtain enough virtual storage below 16 MB to format or access those disks. For more information, see "ACCESS" on page 30.

Responses

```
DMS603R  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' or '1'. You have indicated 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 ', ' 1' or '1. ', the response is processed as a 'NO' response. Responding 'NO', '0', pressing Enter, or entering a character string other than 'YES' or '1' cancels execution of the FORMAT command.

```
DMS605R  Enter disk label:
```

You have requested a label be written on the disk. Enter a 1-6 character label.

```
DMS705I  Disk remains unchanged
```

The response to message DMS603R was other than 'YES'.

```
DMS732I  nnnn {cylinders|FB-512 blocks}
formatted on mode(vdev)
```

The format operation is complete.

```
DMS733I  Formatting disk mode
```

The disk represented by mode letter 'mode' is being formatted.

```
LABEL VDEV M STAT   CYL TYPE BLKSZ   FILES  BLKS USED-(%) BLKS LEFT  BLK TOTAL
label vdev m  R/W nnnnn type nnnn nnnnnnnn nnnnnnnnnn-%% nnnnnnnnnn nnnnnnnnnn
```

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.

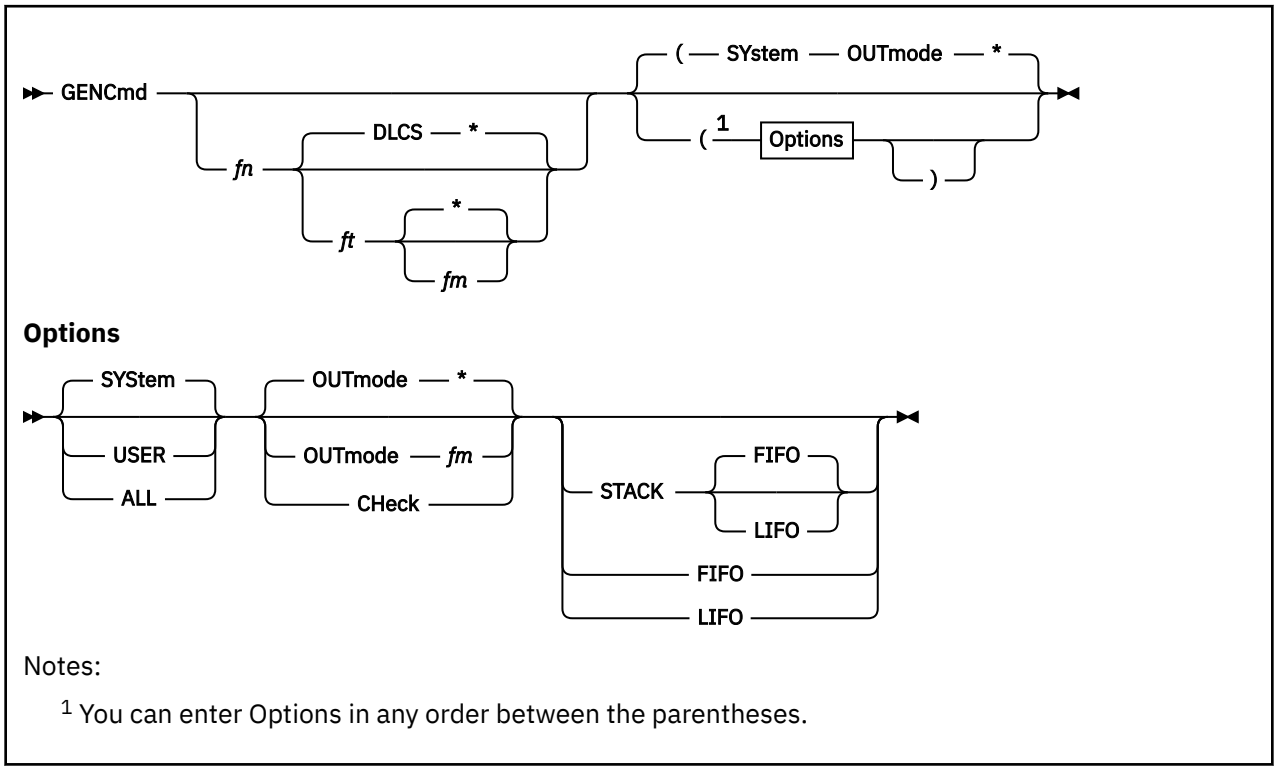
Messages and Return Codes

- DMS003E Invalid option: *option* [RC=24]
- DMS005E No BLKSIZE specified [RC=24]
- DMS017E Invalid device address *vdev* [RC=24]
- DMS028E No device specified [RC=24]
- DMS037E Filemode *mode*[(*vdev*)] is accessed as read/only [RC=36]
- DMS048E Invalid filemode *mode* [RC=24]
- DMS069E Filemode *mode* not accessed [RC=36]
- DMS070E Invalid parameter *parameter* [RC=24]
- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS113S Device *vdev* not attached [RC=100]
- DMS114S Device *vdev* is an unsupported device type, or requested BLKSIZE is not supported for the device [RC=88]
- DMS114S Device *vdev* too large for CMS use [RC=88]
- DMS125S Permanent unit check on disk *mode(vdev)* [RC=100]
- DMS126E Error writing label on disk *mode(vdev)* [RC=100]
- DMS214W Cannot recompute without loss of data; no change [RC=8]
- DMS216E Insufficient blocks on disk to support CMS disk structure [RC=100]
- DMS361E Disk *mode(vdev)* is not a CMS disk [RC=36]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

GENCMD



Notes:

[1] You can enter Options in any order between the parentheses.

Authorization

General User

Purpose

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

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

USER
specifies valid user functions and system functions and their subsets are in the file. You must load user functions as a nucleus extension for GENCMD to process them correctly.

ALL
specifies 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.

OUTmode
specifies the file mode to which the converted output files are written. The *fm* can be any read/write disk or directory 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.

GENCMD always renames the text decks 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.

- System Parser and Synonym Files
  - *xxx*SPA*cc* TEXT
  - *xxx*SSY*cc* TEXT
- User Parser and Synonym Files
  - *xxx*UPA*cc* TEXT
  - *xxx*USY*cc* TEXT

Where:

xxx
identifies the three-character application ID.

S or U
identifies whether the output file is a SYSTEM file or USER file.

PA
identifies the file contains command syntax information.

SY
identifies the file contains command name translations and synonyms.

cc
identifies the 1-2 character country code for the national language you are using.

Note: These utility files:

- COMMANDS CMSUT1

- COMMANDS CMSUT2
- COMMANDS ASSEMBLE
- COMMANDS TEXT

are temporarily created to hold the converted data before it is placed in the output files.

Usage Notes

1. For more information, see *z/VM: CMS Application Development Guide*.
2. When editing a file with DLCS statements, you can issue GENCMD 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 GENCMD from an EXEC program, you should precede it with the EXEC command; that is, specify

   ```
   exec gencmd
   ```

9. If the language identifier specified in the DLCS statement is not one of the valid abbreviations as found in the VMFNLS LANGLIST file, the seventh and eighth characters of the file name being processed are used as the country code.

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:

```
gencmd 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:

```
: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:

```
gencmd test2 dlcs (system
```

The output files are DMSUPAB TEXT and DMSUSYB TEXT.

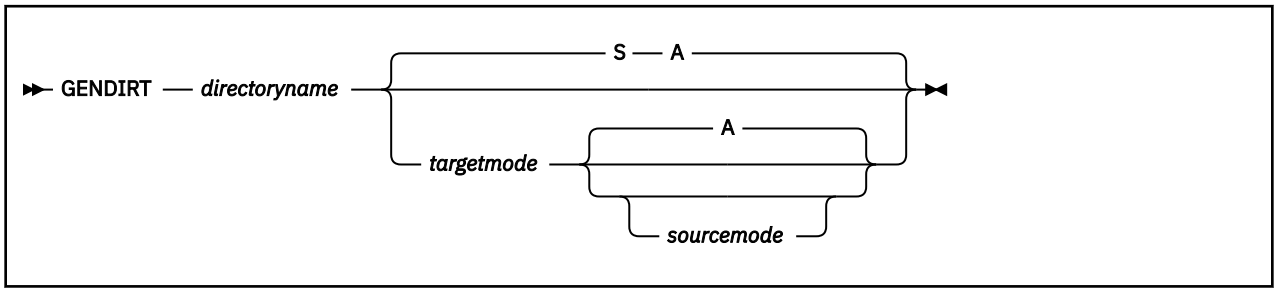
Messages and Return Codes

- DMS002E File *fn ft fm* not found [RC=28]
- DMS006E No read/write filemode accessed [RC=36]
- DMS069E Filemode *fm* not accessed [RC=36]
- DMS396E Maximum number of command table entries exceeded [RC=32]
- DMS639E Error in *routine* routine; return code was *retcode* [RC=256]
- DMS946E XEDIT is not active. Specify a file name. [RC=40]
- DMS947E Line *line*: *syntax error* [RC=8]
- DMS948E Line *line*: *syntax error* [RC=8]
- DMS949E Line *line*: *syntax error* [RC=8]
- DMS950I Conversion of *fn ft fm* [from XEDIT] complete
- DMS950I No errors found in *fn ft fm* [from XEDIT]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411
Errors in the Shared File System	“File Pool Server Messages” on page 1414
Errors in copying a file	“Messages and Return Codes” on page 90

GENDIRT



Authorization

General User

Purpose

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.

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 disk accessed as *sourcemode* (default is A) or a read-only extension of it, because 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.

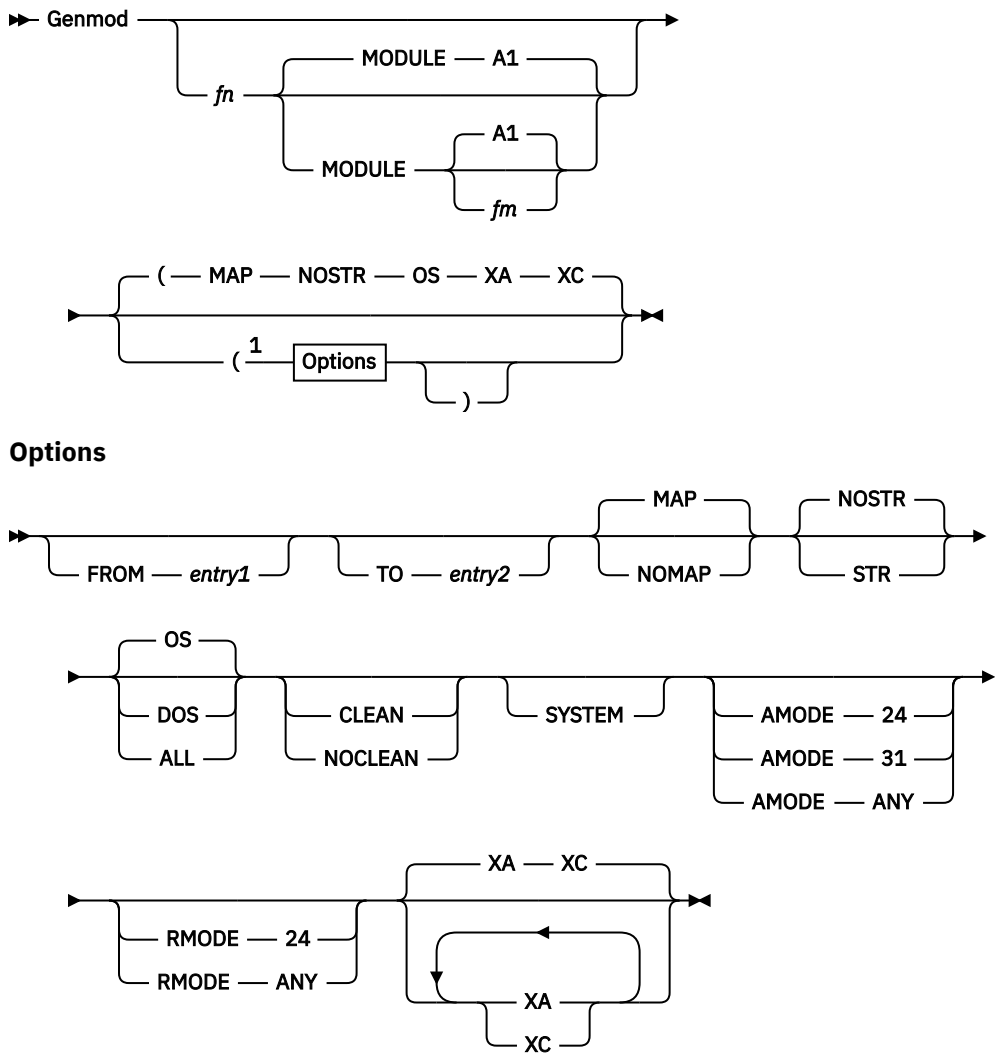
sourcemode
is the mode of the disk that contains the modules or files when the GENDIRT command is issued. If not specified, 'A' is the default.

Note: For information on creating auxiliary directories and for further requirements for using the targetmode option, see *z/VM: CMS Application Development Guide for Assembler*.

Messages and Return Codes

- DMS002W File *fn ft [fm]* not found [RC=4 or 28]
- DMS021E Entry point *name* not found [RC=40]
- DMS022E No directory name specified [RC=24 or 28]
- DMS070E Invalid parameter *parameter* [RC=24]
- DMS1211W FST for file *fn ft fm* not copied [RC=4]
- DMS1264E Filemode *fm* is not associated with a minidisk [RC=4]

GENMOD



Notes:

[1] You can enter Options in any order between the parentheses.

Authorization

General User

Purpose

Use the GENMOD command to create both relocatable and nonrelocatable MODULE files on a disk or directory.

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.

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

FROM *entry1*
specifies an entry point or a control section name that represents the starting virtual storage location from which a MODULE is generated. For more information, see Usage Note "12" on page 355.

TO *entry2*
specifies an entry point or a control section name that represents the ending virtual storage location from which a MODULE is generated. For more information, see Usage Note "12" on page 355.

MAP
copies system loader table entries for the generated module. The record can contain up to 2730 map entries. You can issue the MODMAP command to display the module map. The default is MAP.

Using this option with the NOPRES option on the LOADMOD command deletes previously loaded non-OS programs and causes the loader tables to be rewritten with the map information which was saved when the module was generated.

NOMAP
specifies a module map is not to be contained in the MODULE file.

Note: When NOMAP is specified, the information produced is not sufficient for the START command to execute properly. Therefore, if a module is generated with the NOMAP option, that module cannot later be loaded and started with the CMS LOADMOD and START commands. 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 with the NOPRES option on the LOADMOD command. The default is NOSTR.

STR
deletes previously loaded non-OS programs when used with the NOPRES option on the LOADMOD command.

OS
indicates the program may contain OS macros and, therefore, should be executed only when CMS/DOS is not active. The default is OS.

DOS
indicates the program contains VSE macros; CMS/DOS must be active (that is, SET DOS ON must have been previously invoked) for this program to execute. For more information, see Usage Note "2" on page 354.

ALL
indicates the program:

- Contains CMS macros and must be capable of running regardless of whether CMS/DOS is active
- 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 the module is to be cleaned from storage at the end of its execution. The default is CLEAN, for the relocatable modules.

NOCLEAN
indicates the module is to remain in storage until end-of-command (Ready;). NOCLEAN can be specified for either relocatable or nonrelocatable modules. The default is NOCLEAN, for nonrelocatable 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.

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. If RMODE was specified as ANY in the LOAD process, however, the AMODE will be given a value of 31. This is done to prevent a situation where a module is called from an exec and is given AMODE 24, because REXX and its interpreter can run AMODE 24. With RMODE specified as ANY, however, the program can be loaded outside the 24-bit addressing scheme specified by AMODE 24. 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.

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.

RMODE
specifies the location in virtual storage where the loaded MODULE is to reside.

The RMODE defined by this option overrides the RMODE determined by the LOAD process if the module being generated is relocatable. This means 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 nonrelocatable. This means you did not specify the RLDSAVE option on the LOAD or INCLUDE command used for the MODULE file.

When you load a nonrelocatable 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 16 MB line, overriding the RMODE determined by the LOAD process.

ANY
The load may reside above or below the 16 MB line, overriding the RMODE determined by the LOAD process.

XA
specifies this module can execute in ESA, XA, and XC virtual machines.

XC
when specified as the only architecture attribute, specifies this module can execute only in XC virtual machines.

Note: If you specify neither XA nor XC, no architecture attribute is added to the module, and the module can execute in any kind of virtual machine. If you specify both XA and XC, both architecture attributes are added to the module, and the module can execute in any kind of virtual machine. This is the default.

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 nonrelocatable load module named TESTPROG MODULE. TESTPROG may now be invoked as a user-written command from the CMS environment or loaded into storage and executed with the LOADMOD and START commands.
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. The AMODE/RMODE for the module are set to 24.
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 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 the Usage Notes of, “LOAD” on page 471. 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. A MODULE file generated by GENMOD can contain relocation information if the RLDSAVE option was specified on any of the LOAD or INCLUDE commands used to create this MODULE file. The file is thus 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.
10. If the HIST option was specified on the LOAD or INCLUDE command, the MODULE file may contain history information. For more information, see "INCLUDE" on page 417 and "LOAD" on page 471.
11. Do not XEDIT and then FILE a variable format MODULE file. This deletes any trailing blanks in records, which causes unpredictable results at module execution time. If this occurs, you must regenerate the MODULE.
12. The FROM option is still required when a partial module would otherwise be written as a consequence of Usage Note "7" on page 354. When module relocation information is being saved (using the RLDSAVE option of the LOAD and INCLUDE commands), the FROM option must specify the first byte of the program; it is not optional when Usage Note "7" on page 354 applies. TO should not be specified. If TO or FROM is specified in other ways in conjunction with RLDSAVE, the results will be unpredictable.
13. The SET GEN370 command can be used to allow modules generated with the GENMOD 370 option (no longer supported) to execute in ESA, XA, or XC virtual machines. It is likely the CP SET 370ACCOM ON command will also be necessary for the module to execute.

Examples

Generation of a Nonrelocatable Module

Suppose the TEXT files you specified in the LOAD and INCLUDE commands resulted in the load residing below 16 MB. 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 the executable code functions properly.

Messages and Return Codes

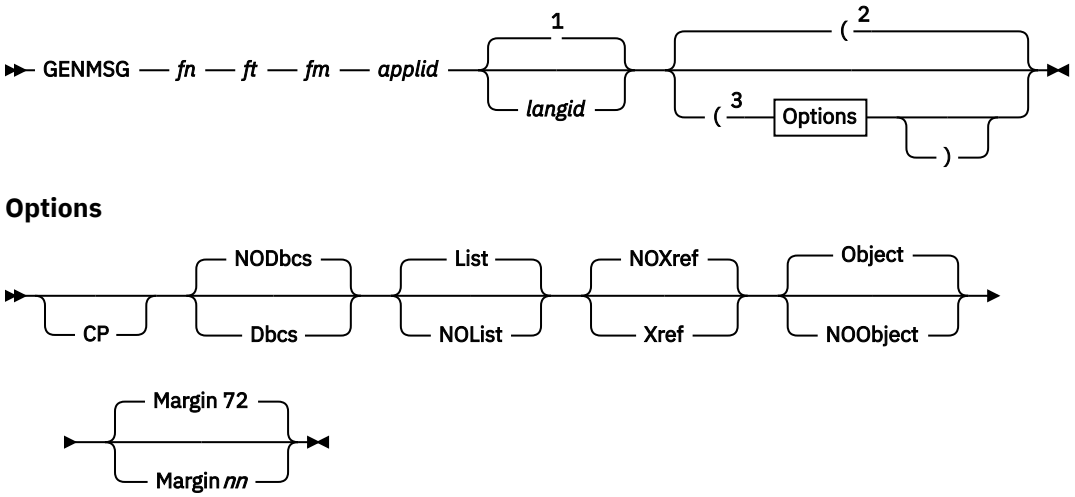
- DMS003E Invalid option: *option* [RC=24]
- DMS005E No *option* specified [RC=24]
- DMS018E No load map available [RC=40]
- DMS021E Entry point *name* not found [RC=40]
- DMS032E Invalid filetype *ft* [RC=24]
- DMS037E Filemode *mode*[(*vdev*)] is accessed as read/only [RC=36]
- DMS040E No files loaded [RC=40]
- DMS069E Filemode *mode* not accessed [RC=36]
- DMS070E Invalid parameter *parameter* [RC=24]
- DMS084E The length of the module to be generated is non-positive. GENMOD terminated. [RC=24]
- DMS105S Error *nn* writing file *fn ft fm* on disk [RC=31 | 55 | 70 | 76 | 99 | 100]
- DMS257T Internal system error at address *addr* (offset *offset*)
- DMS810E 370 cannot be specified as an architecture when AMODE is 31. [RC=68]

- DMS811E [AMODE 24|RMODE 24] cannot be specified when module size exceeds 16Mb. *file* not generated. [RC=68]
- DMS943E Invalid AMODE *mode* specified. *file* not generated. [RC=24]
- DMS944E Invalid RMODE *mode* specified. *file* not generated. [RC=24]
- DMS945E AMODE/RMODE values conflict. *file* not *generated*|*loaded*. [RC=68]
- DMS988E Module *file* cannot execute in XC architecture [RC=64]
- DMS1144E Implicit rollback occurred for work unit *workunitid* [RC=31]
- DMS1252T Rollback unsuccessful for file pool *filepoolid*

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in the Shared File System	“File Pool Server Messages” on page 1414
Errors in using a file	“File Error Messages” on page 1418

GENMSG



Notes:

[1] The default is the current language.
[2] The defaults you receive appear above the line in the Options fragment.
[3] You can enter Options in any order between the parentheses.

Authorization

General User

Purpose

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.

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 an 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 1-5 characters long. The default value is the language currently in use.

Options

CP
notes the input file contains messages for CP. The output repository file is modified accordingly.

Dbcs
specifies the input file contains Double-Byte Character Set (DBCS) characters as part of its message text.

NODbcs
specifies the input file does not contain Double-Byte Character Set (DBCS) characters. The default is NODBCS.

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. The default is LIST. For more information, see Usage Note "7" on page 359. For information on creating a list of files that can be saved and used to send multiple files, see the FILELIST command, Usage Note "7" on page 298.

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. The default is NOXREF.

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. The default is OBJECT.

NOObject
does not create an object file.

Margin *nn*
shows message source is taken from columns 1-nn in the input deck. The default is MARGIN 72. For more information on the margin limits, see "Creating Your Own CMS Messages" in *z/VM: CMS Application Development Guide*.

Usage Notes

1. For more information on how to make your own message repository, see *z/VM: CMS Application Development Guide*.
2. If you choose to change a given CMS message, build your own message repository and include the number of the CMS message you want to override. Load your repository as a user repository using the SET LANGUAGE command.

   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, specify the MARGIN *nn* option as MARGIN 63.

7. If the LIST option is specified and the LPP option on the CP SPOOL command is 0 (LPP OFF), the number of lines per page for the listing file will be 60. If the LPP value is greater than 0, the number of lines per page for the listing file will be equal to the LPP value. For more information, see *z/VM: CP Commands and Utilities Reference*.

Examples

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 of MYAUME REPOS A. This repository contains a message with two formats and a message with five formats. To compile your repository, you enter:

```
genmsg myaume repos a mya (margin 63
```

This 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
*                                                        00002000
0001  01   01    E     No filename specified             00003000
0001  02   01    E     No &1 names specified             00004000
0002  01   01    E     File &1 &2 &3 not found           00006000
0002  02   01    E     &1 file &2 not found              00007000
0002  03   01    E     Dataset not found                 00008000
0002  04   01    E     File(s) &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

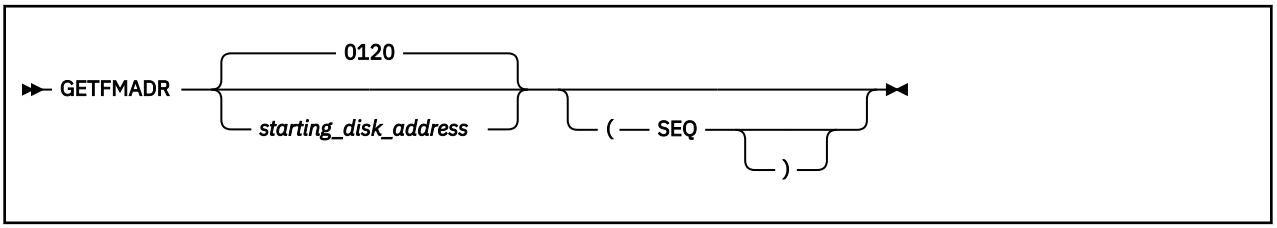
- DMS002E File *fn ft fm* not found [RC=28]
- DMS003E Invalid option: *option* [RC=16]
- DMS006E No read/write filemode accessed [RC=36]
- DMS007E File *fn ft fm* is not fixed, 80-character records [RC=32]
- DMS029E Invalid parameter *parameter* in the option *option* field [RC=24]
- DMS049E Invalid line number *nn* [RC=8]
- DMS054E Incomplete fileid specified [RC=24]
- DMS065E *option* option specified twice [RC=24]
- DMS066E *option1* and *option2* are conflicting options [RC=24]
- DMS104S Error *nn* reading file *fn ft fm* from disk or directory [RC=100]
- DMS105S Error *nn* writing file *fn ft fm* to disk or directory [RC=100]
- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS147E Message not in ascending sequence [RC=8]

- DMS580E Invalid string: unmatched shift-out (SO) and shift-in (SI) [RC=5]
- DMS580E Invalid string: invalid double-byte character(s) [RC=5]
- DMS766I Substitution character is *char*
- DMS767I Number of message number characters to display is *nn*
- DMS768W Invalid substitution character value *char* [RC=4]
- DMS769W Invalid number of message characters value *value* [RC=4]
- DMS770E Invalid application ID *applid* [RC=16]
- DMS771E Invalid message number [RC=8]
- DMS772E Invalid format number [RC=8]
- DMS773E Duplicate message ID *id* [RC=4]
- DMS774E Line numbers for messages are not consecutive [RC=8]
- DMS775W Text too long - 239|229 characters is the maximum allowed [RC=4]
- DMS776I Options used: *list*
- DMS1144E Implicit rollback occurred for work unit *workunitid* [RC=31]
- DMS1229E *fileid* is empty [RC=88]
- DMS1252T Rollback unsuccessful for file pool *filepoolid*
- DMS1262S Error *nn* opening file *fn ft fm*
- DMS2521E GENMSG cannot be performed on empty file *fn ft* [RC=88]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411
Errors in the Shared File System	“File Pool Server Messages” on page 1414
Errors in using a file	“File Error Messages” on page 1418

GETFMADR



Authorization

General User

Purpose

Use the GETFMADR command to stack the first available file mode and an available virtual disk address. An exec can use this information to link and access a minidisk or access a shared file system (SFS) directory without disturbing existing links or accesses.

Operands

starting_disk_address
is the optional virtual disk address, which overrides the default of 0120.

SEQ
causes a sequential search for a free virtual disk address beginning at the specified address or the default address 0120. This option finds the first available virtual disk address without searching through the entire virtual control unit (VCU) range of addresses. The SEQ option overrides the default search for the first free VCU address.

Usage Notes

1. The format of the stacked line is:

   Token
   Content

   1
   * (viewed as a comment by CMS)

   2
   the first free mode letter (or Z if none are free)

   3
   a free virtual disk address

   4
   the current minidisk now accessed as Z, or 0000 if an SFS directory is now accessed as Z

2. GETFMADR without the SEQ option will search for a free virtual disk address beginning at the address you specify, or 0120 by default. It will return the starting address of the first free VCU or, if there are not any free VCUs, the first free address it finds. A VCU is free if there are no devices in the range of 16 addresses, xxx0 - xxxF, the VCU covers.
3. GETFMADR with the SEQ option will return the starting address of the first free address it finds.
4. The stacked line's fourth token is present only if all modes (including Z) are in use. Therefore, check token 4 to determine if there are any free file modes.

Messages and Return Codes

System messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

Return codes:

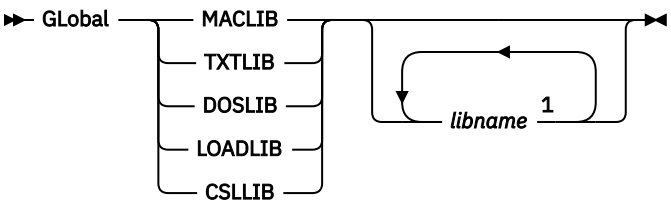
RC
Meaning

0
Command complete—no errors

xx
where xx is RC from PARSECMD or CMSCALL

100
Explanation complete (when ‘?‘ specified)

GLOBAL



Notes:

[1] You can specify a maximum of 63 libnames.

Authorization

General User

Purpose

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.

Operands

MACLIB
precedes the specification of macro libraries 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, when the LKED command is issued, or when a dynamic load occurs (that is, when an OS SVC 8 is issued).

Note: Subroutines called by dynamic load should contain only VCONs resolved within the same text library member, or 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.

LOADLIB
precedes the specification of load module libraries to be searched for a module 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 used by the RTNLOAD command to locate CSL routines. Saved segments are searched first for these libraries, then accessed disks and directories.

Note: VMLIB and VMMTLIB are the names of the callable services libraries supplied with z/VM. They are always implicitly in the search order — if the CSLLIB operand is specified without any library names, VMLIB and VMMTLIB are still available. If VMLIB and VMMTLIB are specified, they are searched in the order given; if they are not specified, they are searched last.

libname...
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 choose 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, because 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 manipulating CMS libraries, see “CSLGEN” on page 111, “DOSLIB” on page 198, “LOADLIB” on page 487, “MACLIB” on page 495, and “TXTLIB” on page 1085.
6. If you want to replace a macro library name with a set of substitute library names, use the SET MACLSUBS command before entering the GLOBAL command.
7. You may receive the DMSGLB108S message even though you specify less than the maximum of 63 library names. This message results because a library name may be replaced by more than one substitute library name if the SET MACLSUBS command was entered.
8. The global list determines the search order of libraries. Libraries can be grouped and referenced by a common DDNAME with the FILEDEF (CONCAT option. For more information on the CONCAT option for the FILEDEF command, see Usage Notes for the CONCAT Option.

Examples

To specify you want the DMSGPI and OSMACRO macro libraries searched when processing CMS commands, you would enter:

```
global maclib dmsgpi osmacro
```

If you enter

```
global csllib userlib1 userlib2
```

the CSL search order becomes: USERLIB1 USERLIB2 VMLIB, with VMLIB and VMMTLIB implicitly included at the end of the search order. Issuing a

```
query csllib
```

will display

```
CSLLIB = USERLIB1 USERLIB2
```

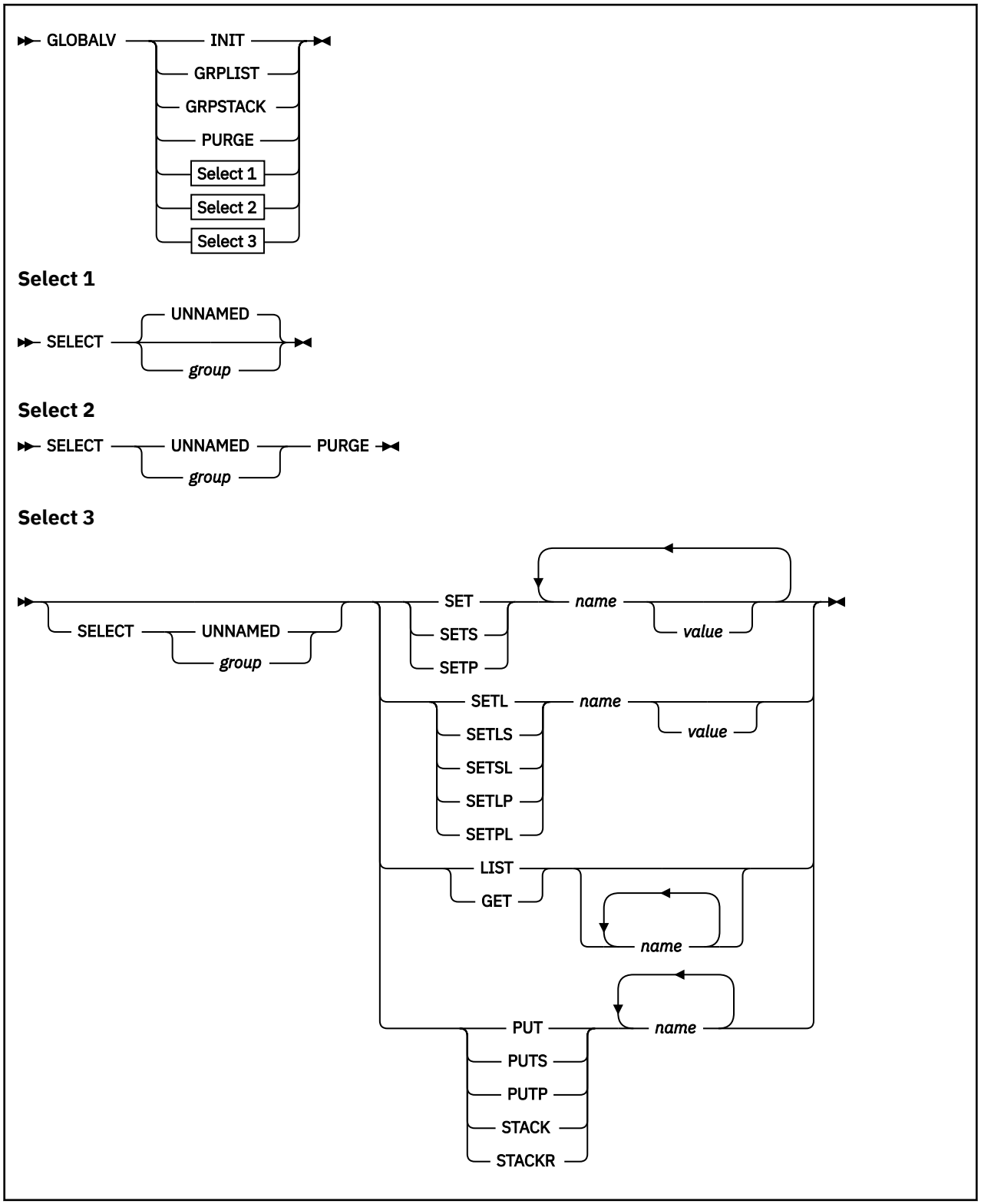
Messages and Return Codes

- DMS002W File *fn ft [fm]* not found [RC=4 or 28]
- DMS014E Invalid function *function* [RC=24]
- DMS047E No function specified [RC=24]
- DMS056E File *fn ft [fm]* contains invalid record formats [RC=32]
- DMS104S Error *nn* reading file *fn ft fm* from disk or directory [RC=31|55|70|76|99|100]
- DMS108S More than *nn* libraries specified [RC=88]
- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS257T Internal system error at address *addr* (offset *offset*)
- DMS1144E Implicit rollback occurred for work unit *workunitid* [RC=31]
- DMS1252T Rollback unsuccessful for file pool *filepoolid*

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in the Shared File System	“File Pool Server Messages” on page 1414
Errors in using a file	“File Error Messages” on page 1418

GLOBALV



Note: Although this command (except for the GET and PUT options) may be used in CMS execs, it is designed for use with REXX or EXEC 2 execs. For more information on the restrictions and precautions, see Usage Notes for CMS Execs.

Authorization

General User

Purpose

The GLOBALV (GLOBAL Variables) command lets several execs share a common set of values, and also retain them (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 groups 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 or variables from the group or groups 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 or groups", 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 reinitialized.

Retaining

When variables are defined or changed, the user decides whether the variables or changes are to last:

- For the current IPL only
- Throughout an entire session (generally, from LOGON to LOGOFF)
- Permanently, for example, 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 or variables in storage. The INITIAL file is generally 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 (through editing). It accommodates variables whose names and values do not exceed 8 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. For more information about fixed and variable formats, see Usage Note “1” on page 370.

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.

Operands

INIT
: allocates and initializes global variable or variables 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, 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 or files 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. For more information, see Usage Notes "2" on page 370 and "3" on page 371.

After initializing free-storage from the LASTING GLOBALV file, the file is rewritten to eliminate multiple definitions for any variable names and any null (zero length) value assignments.

The global variables in free storage are required by all other GLOBALV functions. Therefore, GLOBALV INIT is performed automatically if not explicitly requested before other GLOBALV requests.

GRPLIST
displays a list of all existing global variable groups (including those created by use of the DEFAULTS and EXECUTE commands).

GRPSTACK
places the names of all 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.

Note: This includes those groups created by use of the DEFAULTS and EXECUTE commands.

PURGE
causes all global variables in free storage to be released.

Note: This includes those groups created by use of the DEFAULTS and EXECUTE commands.

SELECT
group
UNNAMED
identifies the global variable group that 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. For the uses of GLOBALV SELECT, see "Examples" on page 371.

SELECT Sub-functions

PURGE
Causes the variables in storage to be cleared. The group itself is not purged.

Attention: If used without SELECT, causes a GLOBALV PURGE of all global variable or variables in storage.

SET, SETS, SETP
assigns the value "value" to the variable "name", and so forth for however many values you have specified. Because SET fields are delimited by blanks, the values cannot contain any blanks. (Use the SETL subfunction for such values.) If no value is specified, the value is assumed to be null.

SET adds the assignments 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 or

replaces the assignments in the selected or default group and appends it to the SESSION GLOBALV disk file. SETP adds/replaces the assignments in the selected or default group and appends it to the LASTING GLOBALV file.

For more information on the CMS exec users, see 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 or replaces the assignment in the selected or default group and appends it to the SESSION GLOBALV file. SETSL is equivalent to SETLS. SETLP adds or replaces the assignment in the selected or default group and appends it to the LASTING GLOBALV file. SETPL is equivalent to SETLP.

For more information on the CMS exec users, see Usage Notes for CMS Execs.

LIST
displays a list of the specified variable name or names and their associated values. If no name is specified, all variables in the selected or default group are listed.

GET
assigns values from the specified or default global variable group to the specified EXEC 2 or REXX 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 REXX exec, and are subject to the rules of the interpreter being used. For more information, see the “Usage Notes” on page 370.

PUT, PUTS, PUTP
determines the value of the EXEC 2 or REXX variable specified in *name*, 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 assignments in the selected or default global variable group in storage. PUTS does the same, but appends the group to the SESSION 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 REXX exec, and are subject to the rules of the interpreter being used.

STACK
places the values associated with the specified variable name or names, 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. For more information, see Example 2. 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 the REXX exec to 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, and so forth. For more information, see Example 1. 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.

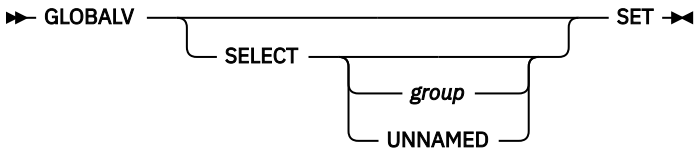
For more information on the CMS exec users, see Usage Notes for CMS Execs.

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 8-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 cannot 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. This is also true for variables in storage.):

```
|------------|----|---------------|----|---------------|
| group name | f1 | variable name | f2 | variable name |
|------------|----|---------------|----|---------------|
0            8    9               n   n+1
```

group name
identifies the group for grouping the variable



f1
defines the actual length of the variable name field immediately following.

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: The GLOBALV command processor rewrites this file, during initialization, it will use variable format.

Probably the easiest way to create GLOBALV files is to let the GLOBALV command processor do the work. Create an EXEC file containing the appropriate GLOBALV ... SETS or SETL commands. Then when the exec is invoked, the GLOBALV command processor will build the file 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 variables 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
   ```

   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 command line
   - An XEDIT macro without an &COMMAND preceding it
   - A REXX 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 REXX/VM, 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 characters) except blanks are permitted following a valid REXX stem. In this case, blanks are treated as delimiters, and therefore are not valid.
6. SVC 202 or CMSCALL, the extended plist, must be in READ/WRITE storage. The extended plist is uppercased by the system translate table if a user's translate table is not available.

Usage Notes for CMS Execs

1. When defining values using GLOBALV's SELECT SET subfunction, 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 SETL subfunction. 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:

   ```
   DMS072E  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:

```
GLOBALV SET DEPT 222            |
 (SELECT phrase is omitted.)    | UNNAMED
                                | Group
Effect:                         |
   The value "222" is assigned  |
   to variable name "DEPT" in   |    .
   the default global variable  |    .
   group "UNNAMED".             | DEPT 222
-------------------------------------------------------
GLOBALV SELECT TABA             |
                                |
Effect:                         |
  The default global variable   |
  group for subsequent SELECT   |
  sub-functions is set to       |
  "TABA"                        |
-------------------------------------------------------
GLOBALV SET EMP 8888 MONTH MAY  |
                                | TABA
Effect:                         |
  The value "8888" is assigned  | Group
  to the variable name "EMP"    |
  and the value "MAY" is as-    |   .
  signed to the variable name   |   .
  "MONTH" in the default group  | EMP 8888
  "TABA".                       | MONTH MAY
-------------------------------------------------------
GLOBALV SELECT UNNAMED SET YEAR | UNNAMED
1982                            | UNNAMED
                                | Group
                                |
 Effect:                        |
   The value "1982" is assigned |
   to the variable name "YEAR"  |    .
   in the global variable group |    .
   "UNNAMED".  The default set- | DEPT 222
   ting is not changed.         | YEAR 1982
-------------------------------------------------------
GLOBALV SETS YEAR 1981          |           SESSION
                                | TABA      GLOBALV
Effect:                         | Group       File
  The value "1981" is assigned  |
  to the variable name "YEAR"   |
  in the default global vari-   |   .           .
  able group "TABA" and the     |   .           .
  assignment is entered into    | EMP 8888   YEAR 1981
  the SESSION GLOBALV file.     | MONTH MAY
                                | YEAR 1981
-------------------------------------------------------
```

```
GLOBALV STACK YEAR DEPT         |      Stack
                                | Before After
Effect:                         |
  Places the value associated   |
  with variable name "YEAR"     | Next line to read:
  "YEAR" from group "TABA"      |
  onto the stack.  Since        | ABC   1981
  the variable "DEPT" is not    | XYZ   (null line)
  defined in global variable    |       ABC
  group "TABA", a null line     |       XYZ
  is stacked.                   |
-------------------------------------------------------
GLOBALV SELECT UNNAMED STACKR   | Stack After
YEAR DEPT                       |
                                |
Effect:                         |
  Places a "&READ 002" control  | Next line to read:
  statement, and two literal    |
  assignment statements, de-    | &READ 002
  fining the variable name      | &YEAR = LITERAL OF 1982
  "YEAR" and the variable name  | &DEPT = LITERAL OF 222
  "DEPT" with their associate   | 1981
  values from global variable   | (null line)
  group "UNNAMED", onto the     | ABC
  stack.                        | XYZ
```

Example 2:

The effect of the following request, which names 3 variables:

```
                                |
GLOBALV SELECT TABA STACK EMP   | Stack After
MONTH YEAR                      |
                                | 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             | Stack After
GLOBALV STACK YEAR              |
                                | 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

- DMS047E No function specified [RC=24]
- DMS104S Error *nn* reading file *fn* GLOBALV A from disk or directory [RC=*nn*000]
- DMS109S Virtual storage capacity exceeded [RC=400041]
- DMS257T Internal system error at address *addr* (offset *offset*)
- DMS618E NUCEXT failed [RC=*nn*]
- DMS622E Insufficient free storage; no table made [RC=*rc*]
- DMS628E Invalid GLOBALV function *function* [RC=4]
- DMS631E *function* can only be executed from an EXEC-2 or REXX EXEC [or as a CMS command] [RC=*rc*]
- DMS649E Extraneous parameter *parameter* [RC=24]
- DMS1144E Implicit rollback occurred for work unit *workunitid* [RC=31]
- DMS1252T Rollback unsuccessful for file pool *filepoolid*

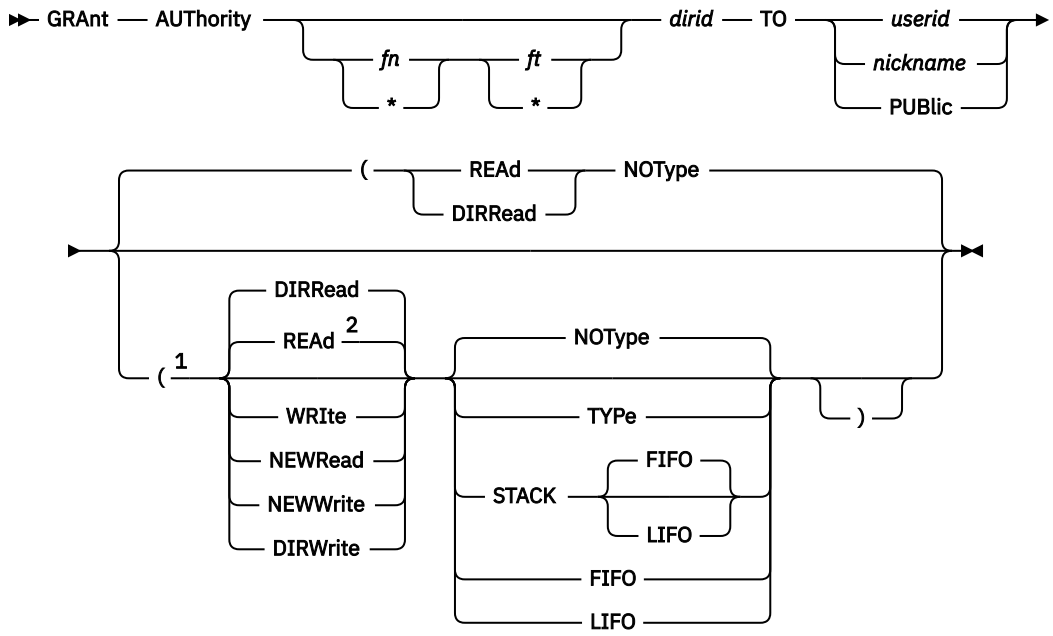
Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in the Shared File System	“File Pool Server Messages” on page 1414
Errors in using a file	“File Error Messages” on page 1418
Errors in copying a file	“Messages and Return Codes” on page 90
Errors in erasing a file	“Messages and Return Codes” on page 217

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
nnn|nnn	
000 ...	Function completed successfully.
001	Truncation to the maximum length of either 255 bytes (for data being appended to a variable length file) or 8 bytes (for data being appended to a fixed length file) occurred for variable name or variable value.
004	Not valid function/sub-function; or a not valid 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 or variables 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. See FSWRITE macro for meaning of "nn".
5nn	EXECCOMM failed with return code "nn". For the meaning of "nn", see the EXECCOMM facility in *z/VM: REXX/VM Reference*.
-5nn	EXECCOMM failed with return code "-nn". For the meaning of "-nn", see the EXECCOMM facility in *z/VM: REXX/VM Reference*.
1nn 000	I/O error reading GLOBALV type files from user's disk or directory accessed as A. No global variable in storage created. See 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. See FSWRITE macro for meaning of "nn".
000	Function completed successfully.
1nn	I/O error appending newly defined variable or variables 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. See 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. For the meaning of "nn", see “RENAME” on page 822.
000	Function completed successfully.
008	Error return from ATTN. Stacking suspended.
400 041	Error occurred when GLOBALV attempted to allocate storage for a work area. GLOBALV initialization functions could not proceed.

GRANT AUTHORITY



Notes:

[1] You can enter Options in any order between the parentheses.

[2] For a file control directory, some option combinations are available: READ NEWREAD, READ NEWWRITE, WRITE NEWREAD, and WRITE NEWWRITE.

Authorization

General User

Purpose

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.

Operands

fn ft
identifies the file for which you are granting authority. Special characters (* or %) can be used to designate a set of files, providing you have appropriate authority for the directory specified by *dirid*.

For more information on *dirid*, see "Naming Shared File System (SFS) Directories" on page 6. For more on using special characters to do pattern matching, see "Using Pattern Matching to Specify Sets of Files" on page 14.

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. You can also use a file mode for the *dirid* if the directory is already accessed. If you have granted a user authority to multiple files in a directory, you should generally grant them authority to the directory, so the grantee can use pattern matching with special characters when issuing commands against it. For more information on the *dirid*, see "Naming Shared File System (SFS) Directories" on page 6.

TO *userid*
TO *nickname*
specifies the name or set of names to whom you are giving read or write authority. User IDs cannot begin with a plus (+) or a minus (–) or contain a colon (:) or a period (.). Nicknames can refer to a group of users. For more information, see "NAMES" on page 535.

TO PUBlic
indicates all users that can connect to the file pool are given authority.

Options

REAd
gives the user READ authority on a file or directory.

READ authority allows the user to create SHARE locks on a file or files, or on the directory. READ authority applies only to file control directories and the files within them.

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. READ authority applies only to file control directories and the files within them. For more information, see "ACCESS" on page 30.

WRIte
gives the user all the privileges of READ authority, plus:

- For a file, the authority to modify, rename, or erase the file. To erase or rename a base file, you must have WRITE authority on the file and to the directory where it resides. To erase or rename an alias, you must have WRITE authority on the directory containing 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.
- For a file or for a directory, the authority to create SHARE, EXCLUSIVE, or UPDATE locks.

Write authority applies only to file control directories and the files within them.

NEWRead
indicates the user (or users) will automatically receive READ authority for any new base files added to the directory. When NEWREAD authority is granted to a user, authorizations to existing files in the directory are not affected. When an alias is added to the directory, the authority to the base file is not affected.

The NEWREAD option is valid only for file control directories. (The *fn ft* operands must be omitted.)

NEWWrite
indicates the user (or users) will automatically receive WRITE authority for any new base files added to the directory. When NEWWRITE authority is granted to a user, authorizations to existing files in the directory are not affected. When an alias is added to the directory, the authority to the base file is not affected.

The NEWWRITE option is valid only for file control directories. (The *fn ft* operands must be omitted.)

DIRRead
indicates the user (or users) will receive directory control read (DIRREAD) authority on the specified directory. This is the default for directory control directories.

DIRREAD authority lets the user:

- Read from the directory
- Read files currently in the directory
- Read any files added to the directory in the future
- Access the directory read-only

DIRREAD authority can be granted only on directories with the directory control (DIRCONTROL) attribute. Omit the *fn ft* operands when granting DIRREAD authority.

DIRWrite
indicates the user (or users) will receive directory control write (DIRWRITE) authority on the specified directory. DIRWRITE authority lets the user:

- Read from and write to the directory
- Read from and write to any files currently in the directory
- Read from and write to any files added to the directory in the future
- Access the directory read-only or read/write
- Create UPDATE locks on files or on the directory

DIRWRITE authority can be granted only on directories with the directory control (DIRCONTROL) attribute. Omit the *fn ft* operands when granting DIRWRITE authority.

TYPe
displays the affected file(s) or directory at the terminal.

NOType
suppresses the display of affected file(s) or directory. The default is NOType.

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

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

LIFO
specifies the output 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. The authorities you can grant and revoke are:
   - For SFS file control directories:

     READ authority
     WRITE authority (which implies READ authority)
     NEWREAD authority
     NEWWRITE authority (which implies NEWREAD authority)

     Note: NEWREAD and NEWWRITE authority are independent from READ or WRITE authority.
   - For files within file control directories:

     READ authority
     WRITE authority (which implies READ authority)
   - For SFS directory control directories:

     DIRREAD authority
     DIRWRITE authority (which implies DIRREAD authority)
   - For files within directory control directories:

     Not applicable. You cannot grant or revoke authorities on individual files within directory control directories. The ability to read from or write to files in directory control directories is derived from DIRWRITE and DIRREAD authority on the directory in which the file resides.
3. When a directory with the file control (FILECONTROL) attribute is created, or when an existing directory is given the FILECONTROL attribute, the owner has WRITE authority and NEWWRITE authority on the directory.

Note: The default attribute for SFS directories is FILECONTROL.

4. When a directory with the directory control (DIRCONTROL) attribute is created or when an existing directory is given the DIRCONTROL attribute, the owner has DIRWRITE authority.
5. For a file control directory, you may grant two authorities at one time: (READ or WRITE) and (NEWREAD or NEWWRITE). The allowed combinations are:

   READ NEWREAD
   READ NEWWRITE
   WRITE NEWREAD
   WRITE NEWWRITE

   Option combinations imply the combined functions of the two options.

   For example, to grant READ and NEWWRITE authority to user Ira on a file control directory named .TEST, you would enter:

   ```
   grant authority .test to ira (read newwrite
   ```

   When granting authority on files or on directory control directories, only one authority may be specified per command.
6. To upgrade READ, NEWREAD, or DIRREAD authority, reissue the GRANT AUTHORITY command with the appropriate option (WRITE, NEWWRITE, or DIRWRITE).
7. To revoke an authority or to downgrade an authority (from WRITE to READ, for instance), use the REVOKE AUTHORITY command.
8. When you grant authority on an alias, the authority refers to the base file. The alias is just a pointer to a base file. Similarly, when you have authority on a base file, you also have the same authority on any alias of the base file.
9. You can grant authority on a file or directory whether the file or directory is locked except when it is locked EXCLUSIVE by another user.
10. 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.
11. You can grant authority on a closed or opened file.
12. If you choose to grant authority to a user ID of PUBLIC, or any of the abbreviations of the PUBLIC operand (PUB, PUBL, and PUBLI), create a nickname for that user ID using the NAMES command. Use the nickname 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 the user is on another processor, the LOCALID tag must also be specified. For more information, see “NAMES” on page 535.
13. You cannot use GRANT AUTHORITY to grant authority on a file or directory that is ESM (External Security Manager) protected. You must use an ESM command to do this.
14. When you grant authority on a directory, that authority does not propagate to subdirectories of that directory.
15. Granting of authority at a different level than currently exists for a user does not affect the current user access to the directory.
16. You cannot use GRANT AUTHORITY to grant authority to external objects. An external object has no authorities pertaining to it. The remote name in the external object may be queried by anyone with read authority to the parent directory.
17. For more information on using the GRANT AUTHORITY command, see *z/VM: CMS User's Guide*.
18. You can invoke the GRANT AUTHORITY command from the command line, from an exec, or as a function from a program. No error messages are issued if GRANT AUTHORITY 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 REXX exec with ADDRESS COMMAND in effect

Messages and Return Codes

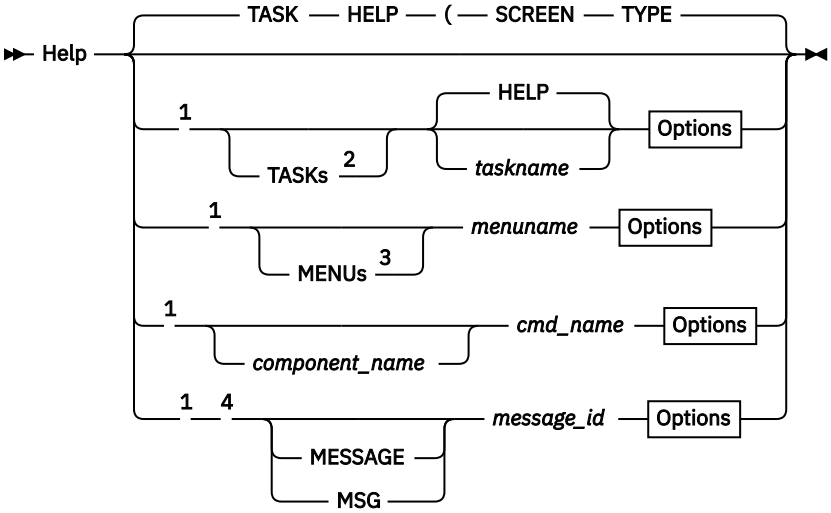
- DMS002E File *fn ft fm* | *dirname* not found [RC=28]
- DMS065E *option* option specified twice [RC=24]
- DMS066E *option1* and *option2* are conflicting options [RC=24]
- DMS069E Filemode *mode* not accessed [RC=36]
- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS149E Userid *userid* not valid [RC=32]
- DMS391E Unexpected operand(s): *operand* [RC=24]
- DMS637E Missing nodeid for the AT operand [RC=24]
- DMS647E Userid not specified for *nickname* in *userid* NAMES file [RC=32]
- DMS647E Localid not specified for *userid* at *node* in *userid* NAMES file [RC=32]
- DMS653E Error executing *command* rc=*nn* [RC=40]
- DMS1163E GRANT AUTHORITY failed for *fn ft dirname* [RC=*nn*]
- DMS1184E File *fn ft fm* | *dirname* not found or you are not authorized for it [RC=28]
- DMS1184E Directory *dirname* not found or you are not authorized for it [RC=28]
- DMS1187E Too many subdirectory levels in *dirname* [RC=24]
- DMS1201E STACK option cannot follow FIFO or LIFO [RC=24]
- DMS1188E Filemode *mode* is not associated with a directory [RC=74]
- DMS1189E Filemode *mode* is associated with a top directory [RC=24]
- DMS1223E There is no default file pool currently defined [RC=40]
- DMS1242E External security in effect for *fn ft fm* | *dirname* GRANT AUTHORITY command cannot be used [RC=88]
- DMS1243W User *userid* already has WRITE authority on *fn ft fm* | *dirname* [RC=4]
- DMS1243W At least one user in the list (*userid*) already has WRITE authority on *fn ft fm* | *dirname* [RC=4]
- DMS1243W User *userid* already has *dirname* [RC=4]
- DMS1243W At least one user in the list (*userid*) already has *dirname* [RC=4]
- DMS1244W {User *userid* | At least one user in the list *userid*} was not granted {READ | WRITE | NEWREAD | NEWWRITE | DIRREAD | DIRWRITE} authority to *fn ft fm* | *dirname* [RC=4]
- DMS1246W Public WRITE authority already granted on *fn ft fm* | *dirname* [RC=4]
- DMS1246W Public WRITE | DIRWRITE | NEWWRITE authority already granted on *dirname* [RC=4]
- DMS1247W Public {READ | WRITE | NEWREAD | NEWWRITE | DIRREAD | DIRWRITE} authority did not previously exist on {*fn ft fm* | *dirname* | *dirname*} [RC=4]
- DMS1247W No users had {READ | WRITE | NEWREAD | NEWWRITE | DIRREAD | DIRWRITE} authority to *fn ft fm* | *dirname* | *dirname* [RC=4]
- DMS1287W You do not own file *fn ft fm* | *dirname* [RC=4]
- DMS1293I You have granted authority to all users of the file pool. [RC=0 | 4]
- DMS2039E DIRREAD | DIRWRITE | NEWREAD | NEWWRITE option cannot be specified on a file [RC=24]
- DMS2039E DIRREAD | DIRWRITE option cannot be specified for a file control directory [RC=24]
- DMS2039E DIRREAD | DIRWRITE | NEWREAD | NEWWRITE option is not supported with the current level of file pool *filepoolid* [RC=88]

- DMS2040E GRANT AUTHORITY cannot be performed on a file in a directory control directory [RC=24]
- DMS2044E READ | NEWREAD | WRITE | NEWWRITE authority cannot be granted on a directory control directory [RC=40]
- DMS2044E READ|WRITE authority cannot be granted on a file within a directory control directory [RC=40]
- DMS2049I DIRREAD authority has been granted to *dirname* for *userid*

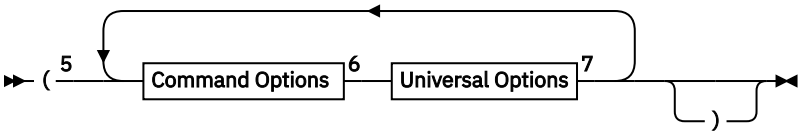
Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411
Errors in the Shared File System	“File Pool Server Messages” on page 1414

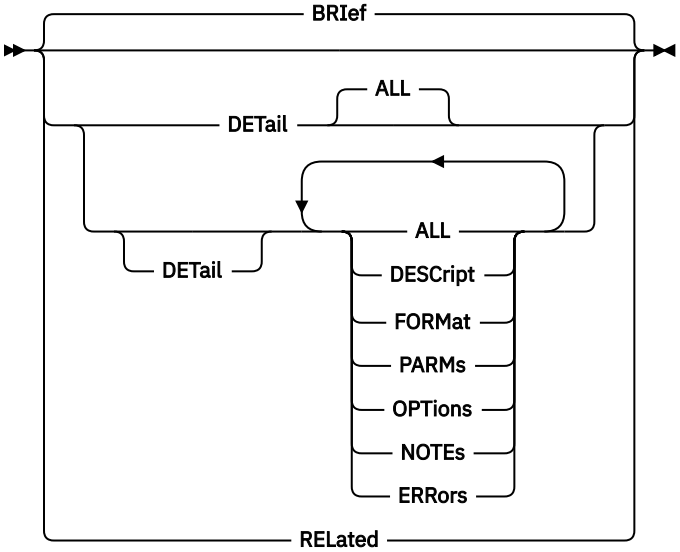
HELP



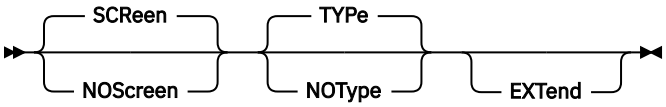
Options



Command Options



Universal Options



Notes:

[1] If you do not specify a type identifier (TASKs, MENUs, MESSAGE, MSG, or *component_name*), then the HELP facility searches among a default list of HELP file types in the following order: HELPCMS,

HELPCP, HELPMSG, HELPMENU, HELPTASK. The HELP facility attempts to match the file name that is specified by the *taskname*, *menuname*, *cmd_name*, or *message_id* operand.
[2] You can specify the TASKs keyword before or after the HELP keyword or the *taskname* operand.
[3] You can specify the MENU keyword before or after the *menuname* operand. You can specify the MENUS keyword only before the *menuname* operand.
[4] For DIRMaint messages (DVHnnnn) you must specify the MSG or MESSAGE keyword before the *message_id* operand.
[5] You can enter options in any order after the opening parenthesis. The HELP command tolerates options that are specified in any order and that are duplicated any number of times. When mutually exclusive options are specified, the last specified option is processed.
[6] The command options affect only HELP files that have a detail information layer. Command options are tolerated but ignored on HELPTASK, HELPMENU, and HELPMSG files.
[7] The universal options are relevant for any HELP file type.

Authorization

General User

Purpose

Use the HELP command to display online information from the z/VM HELP facility. The HELP facility provides information for new and experienced users on the following subjects:

- Tasks.
- Commands and subcommand.
- Messages.
- Routines and callable services.
- Statements.
- Assembly language macros.
- Other products that run under z/VM.

The following types of information are typically found in z/VM HELP files:

- Format of the command, including syntax diagram.
- Authorization that is required to use the command.
- Purpose of the command.
- An explanation of the operands and options that can be used with the command.
- Usage notes.
- Examples.
- Messages that can result from using the command.

The HELP facility can display different layers of information so that you can get an appropriate level of information for your task. For more information, see Options that specify what sections of the HELP file are displayed.

Operands

When the HELP command is specified without any parameters, the HELP facility displays one of two files:

1. HELP HELPMENU. If HELP HELPMENU exists, it is displayed. z/VM HELP does not include a HELP HELPMENU file, but a user can create one.

2. HELP HELPTASK. If HELP HELPMENU does not exist, HELP HELPTASK is displayed. HELP HELPTASK is the main task menu.

TASKs *taskname*

displays the task menu file whose file name is *taskname* (*taskname* HELPTASK).

If a HELPTASK file that has the specified file name cannot be found, the HELP facility searches for synonyms and abbreviations of *taskname* that are defined in an abbreviations file whose file name is TASK HELPABBR (or TASKS HELPABBR). z/VM HELP does not include a TASK HELPABBR file or TASKS HELPABBR file, but users can create HELPABBR files. For more information, see Using Command Abbreviations in *z/VM: CMS User's Guide*.

You can switch the order of the operands. If you specify the *taskname* operand first, then you lose the name-matching capability of the TASK HELPABBR file (or TASKS HELPABBR file), if it exists.

You can specify the TASKS keyword with or without a final S. TASK is valid and TASKS is valid.

If you do not specify *taskname*, the HELP facility displays the high-level task menu (HELP HELPTASK).

If the HELP command does not contain the TASK keyword or the TASKS keyword, then the HELP facility searches in the default components list, which includes files of type HELPTASK. However, the HELPTASK type is fourth in the default search order. If a file with the same name exists with file type HELPCMS, HELPCP, or HELPMENU, then that file is displayed.

Note the one exception to the default search list. HELP HELPTASK is displayed when the following command is entered:

```
help tasks
```

To display the TASKS HELPTASK file, use the following command:

```
help task tasks
```

The task menu can include information files (command files) and menu files (HELPMENU and HELPTASK files). HELPTASK files are organized around user tasks and are called task menu files.

The task menu files are organized in a branching structure of general to specific tasks. The following HELPTASK files are examples of general, more specific, and most specific task menus.

Task File Name
Description

TASKS
z/VM HELP, main panel. A general, high-level task menu.

DUMPS
Help for dump-related commands, subcommands, and utilities. A medium level task menu.

DUMPVIEW
Help for Dump Viewing Facility commands. A low-level task menu with choices that are specific to a single facility.

For another example, enter the following command to display a task menu that lists various tasks for disks.

```
help task disk
```

Some of the disk tasks lead you to more detailed task selection windows. Other disk tasks take you directly to HELP files about the commands that can complete those tasks.

You can create your own task menus. See Creating HELPTASK files in *z/VM: CMS User's Guide*.

MENUs *menuname*

displays the component menu file whose file name is *menuname* (*menuname* HELPMENU).

If a HELPMENU file that has the specified file name cannot be found, the HELP facility searches for synonyms and abbreviations of *menuname* that are defined in an abbreviations file. z/VM provides a MENUS HELPABBR file and users can create and customize HELPABBR files. For more information, see Using Command Abbreviations in *z/VM: CMS User's Guide*.

You can specify the MENUS keyword with or without a final S. When you specify with a final S (MENUS), you can specify only before the *menuname* operand. You can specify the MENU keyword (without a final S) before or after the *menuname* operand. However, if you specify the MENU keyword after the *menuname* operand, then you lose the name-matching capability of abbreviations files.

If the HELP command does not contain the MENU keyword or the MENUS keyword, then the HELP facility searches in the default components list, which includes files of type HELPMENU. However, the HELPMENU type is third in the default search order. If a file with the same name exists with file type HELPCMS or HELPCP, then that file is displayed.

The menu can include information files (command files) and menu files (HELPMENU and HELPTASK files). Compared to HELPTASK files, HELPMENU files are organized around HELP components, which are organized around product components, commands, subcommands, macros, statements, and routines. HELPMENU files are also called component menu files.

The component menu files are organized in a branching structure of general to specific components. The following HELPMENU files illustrate the hierarchy from more general menus to more specific component menus.

Menu File Name
Description

MENUS
The highest level component menu.

CMS
Command files and menus for CMS commands and utilities. A higher level component menu.

XEDIT
Command files and menus for XEDIT. A medium level component menu.

SET
XEDIT command SET subcommand operands. A low level component menu.

To display the highest level component menu, enter the following command:

```
help menu menus
```

From the highest level component menu you can choose the CMS component menu. From the CMS component menu you can choose the XEDIT component menu. From the XEDIT component menu you can choose the SET component menu.

You can create your own component menus. See Creating HELPMENU files in *z/VM: CMS User's Guide*.

component_name cmd_name

displays information in the *cmd_name* HELP file that is in the *component_name* HELP component. In this context, "*cmd_name*" is used generically. The *cmd_name* operand is a HELP file name that describes any function that is started by specifying the name and might be a command, a subcommand, a macro, a statement, or a routine. A *cmd_name* operand is distinguished from a *message_id* operand and from the names of HELPMENU and HELPTASK files.

The HELP facility searches for a HELP file that meets the following criteria:

- The file name matches the *cmd_name* operand.
- The file type is specified by the *component_name* operand, as indicated in Table 19 on page 385.

If the specified file cannot be found, the HELP facility searches for synonyms and abbreviations of *cmd_name* that are defined in an abbreviations file whose file name is *component_name*. For more information, see Using Command Abbreviations in *z/VM: CMS User's Guide*.

You can specify a *component_name* operand that is not the full component name but is only the last 1-4 characters of the HELP file type. For example, the full component name of HELP files for the CP ASSOCIATE command is ASSOCIATE. The file type of the two HELP files in the ASSOCIATE component is HELPASSO. Hence, you could specify a *component_name* operand of ASSO to request HELP information about the ASSOCIATE MESSAGES command:

```
help asso messages
```

When you do not use the full component name, you lose the name-matching capability of the component HELPABBR file. The exception is that if a component name is more than 8 characters, then you gain the name-matching capability of the HELPABBR file if you specify the first 8 characters of the component name. Hence, the following HELP commands yield information tha is in the MESSAGES HELPASSOC file because MSGS is defined as an abbreviation of MESSAGES in the ASSOCIATE HELPABBR file:

```
help associat msgs
help associate msgs
```

In contrast, the following HELP commands yield no HELP information because the *component_name* operand in each command does not identify a HELPABBR file:

```
help asso msgs
help assoc msgs
help associ msgs
help associa msgs
```

Table 19 on page 385 lists the z/VM HELP components and the associated file types and high-level menus.

Table 19. File types, HELP components, and high-level HELP menus

File type	HELP component name	High-level menus	Description
HELPABBR	n/a	n/a	Although HELPABBR files have a unique file type, HELPABBR files are not a HELP component. HELPABBR files are reserved to define abbreviations and synonyms for file names in a HELP component.
HELPADVH	ADVH	ADVH	Directory Maintenance Facility commands
HELPAPPC	APPCVM	APPCVM	APPC/VM macro functions
HELPASSO	ASSOCIATE	ASSOCIAT	CP ASSOCIATE command operands
HELPAVS	AVS	AVS	APPC/VM VTAM® Support commands
HELPBORD	BORDER	BORDER	CMS window border commands.
HELPCMS	CMS	CMS, CMSUTIL, FUNCTION	CMS commands, utilities, and functions
HELPCMSQ	CMSQUERY	CMSQUERY	CMS QUERY command operands
HELPCMSS	CMSSET	CMSSET	CMS SET command operands
HELPCP	CP	CP, CPUTIL, MESSAGE	CP commands and utilities
HELPCPQU	CPQUERY	CPQUERY	CP QUERY command operands
HELPCPSE	CPSET	CPSET	CP SET command operands

Table 19. File types, HELP components, and high-level HELP menus (continued)

File type	HELP component name	High-level menus	Description
HELPCREA	CREATE	CREATE	CMS CREATE command operands
HELPCRR	CRR	CRR	CRR command operands
HELPDEAC	DEACTIVE	DEACTIVE	CP DEACTIVE command operands
HELPDEFI	DEFINE	DEFINE	CP DEFINE command operands
HELPDELE	DELETE	DELETE	Operands of the CMS DELETE and CP DELETE commands
HELPDETA	DETACH	DETACH	CP DETACH command operands
HELPDFSM	DFSMS	DFSMS	DFSMS commands
HELPDIRE	DIRECTORY	DIRECTOR	Directory statements
HELPDISA	DISABLE	DISABLE	CP DISABLE command operands
HELPDISP	DISPLAY	DISPLAY	CP DISPLAY command operands
HELPDRAI	DRAIN	DRAIN	CP DRAIN command operands
HELPDUMP	Several (See note 1): DUMP DUMPSCAN DUMPVIEW	Several: DUMP DUMPSCAN DUMPVIEW	Several: DUMP command operands DUMPSCAN subcommands Dump Viewing Facility commands
HELPEDIT	EDIT	EDIT	EDIT subcommands
HELPENAB	ENABLE	ENABLE	CP ENABLE command operands
HELPEXC2	EXC2	EXEC2	EXEC 2 statements
HELPEXEC	EXEC	EXEC	EXEC statements
HELPFCX	FCX	Several: BASIC FCONTROL FCX REDISP PERFORM	PerfKit for VM subcommands: BASIC mode FCONTROL General panels REDISPLAY mode Performance monitor mode
HELPFILE	FILESERV	FILESERV	File pool server commands (FILESERV command operands)
HELPFLAS	FLASHCOPY	FLASHCOP	CP FLASHCOPY command operands
HELPFREE	FREE	FREE	CP FREE command operands
HELPFTP	FTP	FTP	TCP/IP FTP client (FTP command) subcommands
HELPFTPS	FTPSMSG	FTPSMSG	TCP/IP FTP server administrative subcommands
HELPGIVE	GIVE	GIVE	CP GIVE command operands
HELPGROU	GROUP	GROUP	Group Control System GROUP command and panels

Table 19. File types, HELP components, and high-level HELP menus (continued)

File type	HELP component name	High-level menus	Description
HELPHELP	HELP	HELP	HELP Facility topics
HELPHOLD	HOLD	HOLD	CP HOLD command operands
HELPINCH	INCH	INCH	RSCS Data Interchange Manager commands
HELPINDI	INDICATE	INDICATE	CP INDICATE command operands
HELPIPFO	IPFORMAT	IPFORMAT	TCP/IP IPFORMAT subcommands
HELPIUCV	IUCV	IUCV	IUCV macro functions used with APPC/VM
HELPLDAP	LDAP	LDAP	TCP/IP Lightweight Directory Access Protocol server administrative subcommands
HELPLE	LE	LE	Language Environment commands
HELPLOCA	LOCATE	LOCATE	CP LOCATE command operands
HELPLOGO	LOGO	LOGONID	VM/Pass-Through Facility LOGONID function
HELPMACR	MACROS	MACROS	Preferred CMS macros
HELPMENU	n/a	n/a	Although HELPMENU files have a unique file type, HELPMENU files are not a HELP component. HELPMENU files are reserved as selection files.
HELPMODI	MODIFY	MODIFY	CP MODIFY command operands
HELPMONI	MONITOR	MONITOR	CP MONITOR command operands
HELPMPRO	MPROUTE	MPROUTE	TCP/IP MPROUTE server administrative commands
HELPMSG	MSG	n/a	z/VM messages
HELPMSOC	MSOCKETS	SMAPI (Task menu)	Systems management sockets
HELPNFS	NFS	NFS, REFRESH	TCP/IP NFS server administrative commands, TCP/IP Network File System Refresh commands
HELPNSIN	NSINTER	NSINTER	TCP/IP NSLOOKUP interactive subcommands
HELPNSLO	NSLOOKUP	NSLOOKUP	TCP/IP NSLOOKUP set subcommands
HELPOMAC	OMACRO	OMACRO	OpenExtensions assembler language macros
HELPOPEN	OPENVM	OPENVM	CMS OPENVM commands
HELPOROU	OROUTINE	OROUTINE	OpenExtensions callable services
HELPOSHE	OSHELL	OSHELL	OpenExtensions shell commands
HELPPIPE	PIPE	PIPE	CMS Pipelines stages and pipeline subcommands
HELPPREF	PREFIX	PREFIX	XEDIT prefix subcommands and macros
HELPPSCR	PSCREEN	PSCREEN	CMS physical screen commands
HELPPURG	PURGE	PURGE	CP PURGE command operands
HELPPVM	PVM	PVM	VM/Pass-Through Facility

Table 19. File types, HELP components, and high-level HELP menus (continued)

File type	HELP component name	High-level menus	Description
HELPQUER	QUERY	QUERY	XEDIT QUERY subcommand options
HELPRDEV	RDEVICE	RDEVICE	CP SET RDEVICE command operands
HELPRDVF	RDVF	RDVF	Dump Viewing Facility DUMPSCAN subcommands for RSCS
HELPREXX	REXX	REXX	REXX/VM statements
HELPROUT	ROUTINES	Several: CPIC CPIRR CRRPART CRRR ERRORCHK FILESYSR FPADMINR MULTITSK OTHERR REXXR ROUTINES RSKROUT VMDSR WORKUNIT	Routines: CPI communications CPI resource recovery CRR participation Coord. Resource Recovery CMS error checking and debug CMS file system mgmt (I/O) CMS file pool administration CMS application multitasking Other CMS routines REXX-related CSL routines CMS routines and services CMS Reusable Server Kernel VM data space CMS work unit management
HELPRQUE	RQUERY	RQUERY	RSCS QUERY command operands
HELPRSCS	Several (See note 1): RSCS RSCSAUTH	Several: RSCS RSCSAUTH	Several: RSCS commands RSCSAUTH commands
HELPRSKC	RSKCMDS	RSKCMDS	Reusable Server Kernel commands
HELPRXSO	RXSOCKET	RXSOCKET	REXX Sockets functions
HELPSEGM	SEGMENT	SEGMENT	CMS SEGMENT command operands
HELPSET	SET	SET	XEDIT SET subcommand options
HELPSFSA	SFSADMIN	SFSADMIN	SFS/CRR administrator and operator commands
HELPSFSQ	SFSQUERY	SFSQUERY	SFS/CRR administrator and operator QUERY commands
HELPSMTP	SMTP	SMTP	TCP/IP SMTP server administrative commands
HELPSNMP	SNMP	SNMP	TCP/IP SNMP administrative commands
HELPSPXT	SPXTAPE	SPXTAPE	CP SPXTAPE command operands
HELPSQLD	n/a	n/a	The file type is reserved for DB2® Server for VM
HELPSSLA	SSLADMIN	SSLADMIN	TCP/IP SSL server administrative commands
HELPSTAR	START	START	CP START command operands

Table 19. File types, HELP components, and high-level HELP menus (continued)

File type	HELP component name	High-level menus	Description
HELPSTOR	STORE	STORE	CP STORE command operands
HELPSYSC	SYSCONFIG	SYSCONFI	System configuration statements
HELPTASK	n/a	n/a	Although HELPTASK files have a unique file type, HELPTASK files are not a HELP component. HELPTASK files are reserved as selection files.
HELPTCPI	TCPIP	TCPIP	TCP/IP commands and related functions
HELPTELN	TELNET	TELNET	TCP/IP Telnet protocol client subcommands
HELPTERM	TERMINAL	TERMINAL	CP TERMINAL command operands
HELPTFTP	TFTP	TFTP	TFTP command operands
HELPTRAC	TRACE	TRACE	CP TRACE command operands
HELPTRSO	TRSOURCE	TRSOURCE	CP TRSOURCE command operands
HELPTSAF	TSAF	TSAF	Transparent Services Access Facility commands
HELPUDVH	UDVH	UDVH	Directory Maintenance Facility commands in upper case English
HELPUFTD	UFTD	UFTD	TCP/IP UFT server administrative (UFTD command) subcommands
HELPVARY	VARY	VARY	CP VARY command operands
HELPVMDS	VMDS	VMDS	VM data spaces CP macros
HELPVMDT	VMDT	VMDT	VM Dump Tool toolkit macros and stages (unsupported samples and examples)
HELPVMDU	VMDUMPTL	VMDUMPTL	VM Dump Tool subcommands and macros
HELPVMFI	VMFINS	VMFINS	VMFINS command operands
HELPVMFS	VMFSIM	VMFSIM	VMFSIM command operands
HELPVMSE	VMSESE	VMSES, SERVMGR	VMSES/E commands, SERVMGR command operands
HELPVSCR	VSCREEN	VSCREEN	CMS virtual screen commands
HELPWIND	WINDOW	WINDOW	CMS window commands
HELPXEDI	XEDIT	XEDIT	XEDIT subcommands
HELPXLIN	XLINK	XLINK	CP XLINK command operands

1. Multiple HELP components are associated with this file type. Each component is associated with a unique abbreviations file.

Some HELP components might not be installed on your system. In addition to the z/VM HELP components, you can create your own HELP components. For more information, see Tailoring the HELP facility in *z/VM: CMS User's Guide*.

More information about the relationship between file types, component names, and abbreviations files is available. See HELP Components Definitions and Purpose in *z/VM: CMS User's Guide*.

Default search order

If you omit the *component_name* operand from the HELP command, the HELP facility searches a default list of HELP components. The components are searched in the following order:

1. *cmd_name* HELPCMS
2. *cmd_name* HELPCP
3. *cmd_name* HELPMENU
4. *cmd_name* HELPTASK
5. *cmd_name* HELPMSG

MESSAGE *message_id*
MSG *message_id*

displays the HELP file for a message (*message_id* HELPMSG). The *message_id* can be specified in a 6-12 character format:

```
xxxnnn
xxxnnns
xxxnnnnn
xxxnnnns
xxxmmmnnn
xxxmmmnnns
xxxmmmnnnn
xxxmmmnnnns
```

Where:

xxx
is the product component identifier; for example, DMS for CMS messages, HCP for CP messages. The *message_id* must include the product component identifier.

mmm
is the module identifier.

nnn
nnnn
is the message number. The *message_id* must include the message number.

s
is the message severity code. Specify the severity code for all messages except DIRMaint messages (DVH prefix).

For example, you can specify any of the following *message_id* variations to display the same HELP file:

```
DMS250S
DMS0250S
DMSHLP250S
DMSHLP0250S
```

If the HELP command does not contain the MSG or MESSAGE keyword, the HELP facility searches in the default components list, which includes files of type HELPMSG.

Options

Options for the HELP command are of two categories:

1. Command options
2. Universal options

Command options

Command options specify which sections of the HELP file are displayed. Command options include layering and subsetting options. The layering and subsetting options have a direct correspondence to the sections of a command HELP file. For more information, see Structure of Command HELP files in *z/VM: CMS User's Guide*.

Three *layering* options specify three layers of information:

1. BRIEF
2. DETAIL
3. RELATED

BRIEF, DETAIL, and RELATED are mutually exclusive options. If you specify more than one layering option at a time, the HELP facility ignores all but the last layering option that you specify.

You can display the different layers by specifying options and by pressing PF keys. For more information about using PA and PF keys in the HELP facility, see Using the PA2 and PF2 Keys in *z/VM: CMS User's Guide*. Not all layers occur in all HELP files.

The detail layer of information in a HELP file can contain up to 6 sublayers of information, each of which is specified by one of the *subsetting* options:

1. DESCRIPT
2. FORMAT
3. PARMS
4. OPTIONS
5. NOTES
6. ERRORS

You can specify more than one subsetting option. The ALL subsetting option displays all the detail sublayers.

If the DETAIL layering option is specified or if no layering option is specified, then you can specify subsetting options.

The following options specify the information layer that the HELP facility displays.

BRIef
displays the information that is in the HELP file section that is delimited by the `.CS 0` or `.CS BRIEF` control word. The information might be a description of the specified command, its basic syntax (command without options), an example, and, if applicable, a message that either ALL or RELATED information is available. BRIEF is the default layering option.

If no layering option is specified, then by default the HELP facility displays the brief layer of information.

Note: Although the HELP facility supports the BRIEF option and information layer, z/VM HELP files do not include the brief information layer.

DETail
displays one or more parts of the detail layer of a HELP file. The detail layer is the information in the HELP file sections that are delimited by the `.CS` control words with values 1-6 or the corresponding keyword values. The z/VM HELP files contain detailed command information in those sections. You can filter the detail information that is displayed by specifying one or more subsetting options. See Subsetting Options.

RELated
displays the information that is in the HELP file section that is delimited by the `.CS 7` or `.CS RELATED` control word. The z/VM HELP files contain a menu from which you can select related HELP files.

For example, the following command displays a menu panel from which you can select HELP files about commands that are related to the SENDFILE command.

```
help sendfile (related
```

The task panel might display the following menu choices that are related to the SENDFILE command:

NAMES
NOTE
RDRLIST

RECEIVE
SET MSG
TELL

Subsetting options filter the detail information that is displayed. The following subsetting options are valid:

ALL
displays all sections of the detail layer that are in the HELP file. The ALL option is the default subsetting option.

DESCript
displays the information that is in the section of the HELP file that is delimited by the `.CS 1` or `.CS DESCRIPT` control word. The z/VM HELP files typically contain information that is titled "Purpose" or "Authorization".

FORMat
displays the information that is in the section of the HELP file that is delimited by the `.CS 2` or `.CS FORMAT` control word. The z/VM HELP files typically contain information with a title that matches the command name and contains a syntax diagram.

PARMs
displays the information that is in the section of the HELP file that is delimited by the `.CS 3` or `.CS PARMS` control word. The z/VM HELP files typically contain information that is titled "Operands" or "Parameters" and contains an explanation of the command operands.

OPTions
displays the information that is in the section of the HELP file that is delimited by the `.CS 4` or `.CS OPTIONS` control word. The z/VM HELP files typically contain information that is titled "Options" and contains an explanation of the command options.

NOTEs
displays the information that is in the section of the HELP file that is delimited by the `.CS 5` or `.CS NOTES` control word. The z/VM HELP files typically contain information that is titled "Usage Notes" or "Examples".

ERRors
displays information that is in the section of the HELP file that is delimited by the `.CS 6` or `.CS ERRORS` control word. The z/VM HELP files typically contain information that is titled "Responses" or "Messages".

Universal options
The HELP command uses several other options that control how the information is displayed, whether messages are suppressed, and whether extra HELP components are searched to find a HELP file. The other options do not specify what sections of the HELP file are displayed and are relevant for all HELP files.

SCReen
specifies that the entire screen displays the HELP file. While viewing the help file, you can use PF keys and some XEDIT commands to manipulate the display. This is the default. This option is ignored on a line-oriented terminal.

NOScreen
specifies that the file is displayed without using XEDIT.

TYPe
allows error message DMS254E to be issued. This is the default.

NOType
suppresses the display of error message DMS254E. The option allows you to change the text and placement of the message.

EXTend
uses the HELP search order when you issue the HELP command and specify the HELP component name (*comp_name*). If the file is not found in the specified HELP component, the HELP facility searches for the file name in the default list of HELP components.

The HELP facility searches the default list of HELP components when both the following conditions are true:

- The HELP command does not specify a HELP file type identifier: *component_name* or MENU or TASK keyword.
- The HELP command specifies the HELP file name, which in the syntax diagram is one of the following variable operands: *cmd_name*, *menuname*, or *taskname*.

Usage Notes

1. HELP is always displayed in a window HELPWIN that is showing virtual screen HELPWIN. When displaying BRIEF HELP using full-screen CMS, the CMS window is scrolled up, if required, so 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 are preset to BRIef, ALL, and SCReen. However, the options you specify on the command line when you enter 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 want. For more information, see "DEFAULTS" on page 153.
3. The HELP disk is specified at system generation time by the system support personnel. If the disk is not already accessed, HELP accesses the disk that contains the system HELP files. The HELP disk is accessed by using the last available file mode and remains accessed after HELP completes processing.
4. For commands or statement names that contain special characters, use the special character. For example, if 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 *z/VM: 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. If you specify `CONT` instead, scrolling is continuous to the end of a file.
6. The following screen is an example of the information that is displayed in the following situations:
   - You request a help menu on a line-oriented terminal, for example by issuing the command `HELP XEDIT MENU`.
   - You request a help menu on a display terminal by using the NOSCREEN option, for example by issuing `HELP XEDIT MENU (NOSCREEN`.

```
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'.
&         *Prefix   *QUERY    *SET      ?         :XEDIT    =         Add
ALL       ALter     BAckward  Bottom    CANCEL    CAppend   CDelete   CFirst
Change    CInsert   CLAst     CLocate   CMS       CMSG      CMSXEDIT  COMMAND
COMPress  COpy      COUnt     COVerlay  CP        CReplace  CURsor    DELete
Down      DUPlicat  EMSG      EXPand    EXTract   FILE      Find      FINDUp
FOrward   GET       Help      HEXType   Input     Join      LEft      LOAD
Locate    LOWercas  LPrefix   MACRO     MErge     MODify    MOve      MSG
Next      NFind     NFINDUp   Overlay   PARSE     POWerinp  PREServe  PURge
PUT       PUTD      Query     QUIT      READ      RECover   REFRESH   RENum
REPeat    Replace   RESet     RESTore   RGTLEFT   RIght     SAVE      SCHANGE
SET       SHift     SI        SORT      SOS       SPlit     SPLTJOIN  STAck
STATus   SUPerset   TOP       TRAnsfer  Type      Up        UPPercas  VMFDEOPT
VMFOPT   Xedit
Ready;
```

The following screen is an example of the information that is displayed in the following situations:

- You display a help task on a line-oriented terminal, for example by issuing the command `HELP TASK`.

- You request a help task on a display terminal by using the NOSCREEN option, for example by issuing the command HELP TASK (NOSCREEN.

```
help task (noscreen
For information on one of the following topics, enter
HELP followed by the request information for that topic.

Request             Topic
HELPINFO TASK       HELPINFO  - z/VM HELP Facility
MENUS MENU          MENUS     - z/VM Help menus
TASKS TASK          TASKS     - Basic z/VM tasks
AVS MENU            AVS       - AVS commands
CMS MENU            CMS       - CMS commands
CP MENU             CP        - CP commands
DIRECTOR MENU       DIRECTORY - CP Direcotry Statements
ADVH MENU           DIRMAINT  - DIRMaint commands
ADVH TASK           DIRMAINT  - DIRMaint topics
DUMPS TASK          DUMPS     - Dump commands, subcommands, and utilities
DYNIO TASK          DYNIO     - Dynamic I/O tasks
LE MENU             LE        - Language Environment commands
MACROS MENU         MACROS    - CMS assembler macros (menu)
MACROS TASK         MACROS    - Assembler macros and functions (types)
MESSAGE TASK        MESSAGES  - Messages and codes
OPEN TASK           OPEN      - OpenExtensions services and APIs
PERFKIT TASK        PERFKIT   - Performance Toolkit topics
PIPE MENU           PIPE      - CMS Pipelines built-in programs and commands
QUERYSET TASK       QUERYSET  - QUERY and SET commands and subcommands
ROUTINES MENU       ROUTINES  - CMS callable services (menu)
ROUTINES TASK       ROUTINES  - CMS routines (types)
RSCS MENU           RSCS      - RSCS Networking commands and link parameters
STATEMTS TASK       STATEMTS  - REXX, EXEC 2, and EXEC statements
SUBCOMMD TASK       SUBCMDS   - Subcommand groups, such as XEDIT
SYSCONFIG MENU      SYSCONFIG - CP System Configuration Statements
TCPIP MENU          TCPIP     - TCP/IP commands
TCPIP TASK          TCPIP     - TCP/IP tasks
TSAF MENU           TSAF      - TSAF commands
VMSES MENU          VMSES     - VMSES/E commands
Ready;
```

7. If you view 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 options on the command line to view certain sections of the HELP file. For example, if you are viewing a HELP file and decide 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 is 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 might want to EXECDROP the HELPXED XEDIT macro after using HELP to release the storage.
9. Any data that is not within a conditional section (.cs on/.cs off) is uncontrolled data; all uncontrolled data defaults to the detail layer.
10. If a command HELP file does not contain all information layers and sublayers, the information that is displayed by a specified or default option might be an alternative information layer or sublayer. The following situations can occur:
    - If you specify the BRIEF option and the BRIEF section does not exist in the specified HELP file, then the HELP facility displays message DMSHEL244W and an alternative layer of information. If the DETAIL section exists, then the HELP facility displays the DETAIL section. If the DETAIL section does not exist and the RELATED section exists, then the HELP facility displays the RELATED section.
    - If no layering option is specified and the BRIEF section does not exist in the specified HELP file, then by default the HELP facility displays the DETAIL section.
    - If you specify the DETAIL option and the DETAIL section does not exist in the specified HELP file, then the HELP facility displays message DMSHEL244W and an alternate layer of information. If the

BRIEF section exists, then the HELP facility displays the BRIEF section. If the BRIEF section does not exist and the RELATED section exists, then the HELP facility displays the RELATED section.

- If no layering option is specified and the BRIEF section and the DETAIL sections do not exist in the specified HELP file, then by default the HELP facility displays the RELATED section.
- If you specify the RELATED option and the RELATED section does not exist in the specified HELP file, then the HELP facility returns the following message and displays no sections of the HELP file:

```
DMSHEL243I  Related information is not available.
```

- If any of the sublayers that you specify on the HELP command exist in the specified HELP file, then the HELP facility displays the specified sublayers that exist in the HELP file.
- If none of the sublayers that you specify on the HELP command exist in the specified HELP file, then the HELP facility displays all the sublayers that exist in the HELP file.

11. Some CMS commands are inappropriate to the BRIEF HELP environment. They are:
    - WINDOW DELETE
    - WINDOW DROP
    - WINDOW HIDE

    Issuing any command that alters, drops, hides, or changes the contents of a BRIEF window might cause undesirable results.

    Furthermore, you can use all XEDIT commands while you view the displayed HELP files except the following commands:

    - ALL
    - FILE
    - INPUT
    - MACRO
    - READ
    - REPLACE
    - SET
    - POWERINP
12. The HELP facility processes only the first eight characters of the file name and component name operands (*taskname*, *menuname*, *comp_name*, *cmd_name*, *HELPMENU_filename*, *HELPTASK_filename*). The HELP facility ignores extra characters.
13. Layering and subsetting options interact in the following ways:
    - If you specify the DETAIL option and specify no subsetting options, the ALL subsetting option is the default. All the detail sections are displayed.
    - You can specify the DETAIL option and one or more subsetting options.
    - If you specify any subsetting option and do not specify the BRIEF or RELATED layering options, you don't have to specify the DETAIL option.
    - If several sections are displayed, then they are displayed in the order in which they occur in the HELP file.
      - Sections are not necessarily displayed in the order of the numeric value of the `.CS` word label. Sections are displayed in the order of the numeric value of the `.CS` word label only if they occur in the order of the numeric value of the `.CS` word label in the HELP file.
      - If there are two or more sections of the same type that are separated by a section of another type, then the sections are displayed in the order in which they occur in the HELP file. In other words, sections of the same type are displayed contiguously only if they occur contiguously in the HELP file.
    - If you specify the BRIEF or RELATED layering options, then any subsetting options are ignored.

Examples

Following are some examples of HELP requests.

- To request a HELP file for *message* DMS002E, enter any of the following commands:

```
help dms002e
```

```
help dmshlp002e
```

```
help message dms002e
```

```
help msg dms002e
```

- To request a menu of HELP files for CP commands, issue:

```
help cp menu
```

- To display information about the XEDIT LOCATE subcommand, enter the following command:

```
help xedit locate
```

- To display only description information about the CMS TAPE command, enter either of the following commands:

```
help cms tape (desc
```

```
help tape (desc
```

- If you are viewing the HELP file for the CMS PRINT command and decide that you want to display information about the CMS COPYFILE command, enter the following command:

```
help cms copyfile
```

- To display a menu of HELP files that are related to the CMS SENDFILE command, enter the following command:

```
help cms sendfile (related
```

- You want information about the DUMP command. You guess that the HELP file is in the DUMP component, so you specify the DUMP HELP component on the HELP command. However, you're not sure if DUMP is the right component, so you extend the search to the default HELP components search list by entering the following command:

```
help dump dump (extend
```

  The HELP facility does not find the file in the DUMP component, but extends the search to the default list of HELP components. The HELP facility finds and displays the DUMP HELPCP file, which is in the CP HELP component.

- To request a HELP file for the CMS SET APL command, you enter the following command:

```
help cmsset apl
```

- To display menu choices for CMS macros, enter the following command:

```
help macros
```

  No HELP component is specified, but the MACROS HELPMENU file is found when the HELP facility searches the default HELP components list.

- To display choices for routines, enter the following command:

```
help routines
```

No HELP component is specified, but the ROUTINES HELPMENU file is found when the HELP facility searches the default HELP components list.

Messages and Return Codes

- DMS003E Invalid option: *option* [RC = 24]
- DMS241I Press PF10 for detail information.
- DMS241I Press PF11 to get related information.
- DMS241I Press PF10 for detail information; PF11 to get related information.
- DMS242I This HELP file *fn ft* has not been converted to the current release format or contains an invalid format word.
- DMS243I RELATED information is not available.
- DMS244W Requested HELP section unavailable; *option* option assumed.
- DMS244W Requested HELP subset information is not available; the *option* option has been assumed.
- DMS254E 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]
- DMS355I For related information on this subject, enter MOREHELP (RELATED.
- DMS356I For more detail on this subject, enter MOREHELP.
- DMS529E Subcommand is only valid in display mode
- DMS529E SET *commandword* subcommand is only valid in editing mode
- DMS545E Missing operand(s)
- DMS561E Cursor is not on a valid data field
- DMS586E String not found
- DMS639E Error in *routine* routine; return code was *xx*
- DMS657E Undefined PFkey/PAkey.
- DMS1229E Help file *fileid* is empty [RC=88]

Additional system messages might be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

HELPCONV

Authorization

General User

Purpose

Use the HELPCONV command to convert a specified file into a formatted HELP file, leaving the .CS, .CM, and .MT control words in the file. The output file has the file type $HLP*xxxx*, where *xxxx* is the name of the component to which the file belongs.

Operands

fn
is the file name of the HELP file to be converted.

ft
is the file type of the HELP file to be converted.

fm
is the file mode of the HELP file to be converted. If the file mode is omitted or if an asterisk (*) is coded, all accessed disks and directories are searched.

Usage Notes

1. After using the HELPCONV command on a specified HELP file, you will have two versions of the Help file:

   ***fn* HELPxxxx A1**
   The original file

   ***fn* $HLPxxxx A1**
   The converted file

   You should verify the $HLPxxxx file is formatted the way you want it to appear when displayed by HELP.

   To use the converted file in the HELP command, you must rename the converted file so its file type is HELPxxxx. For example, if you enter

   ```
   helpconv link helpcms
   ```

   you create a converted file with a file ID of LINK $HLPCMS A1. To have the HELP command call the new HELP file, rename the file so the file type is HELPCMS:

   ```
   rename link $hlpcms a link helpcms a
   ```

   Note the converted file has the same name as the original file. Therefore, to rename the converted file, you must first rename the original file, move it to another read/write minidisk or directory, or copy the formatted file to another read/write minidisk or directory, specifying the file type as HELPxxxx.
2. To use Help format words other than .CS, .CM, and .MT (for instance, .BX, .IN, and so forth) in your own HELP files, it's still necessary to use the HELPCONV command. It is also *critical* in these situations to

ensure the proper formatting of MENU and TASK files, that is, the file begins with a .FO OFF format word. Similarly, when you use HELPCONV on a file that contains a RELATED section, ensure you turn off the formatting for the entire RELATED section. For more information on how to process these files, see "Tailoring the HELP Facility" in *z/VM: CMS User's Guide*.

Messages and Return Codes

- DMS002E File not found [RC=28]
- DMS006E No read/write disk accessed [RC=36]
- DMS104S Error *nn* reading file *fn ft fm* from disk or directory [RC=104]
- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS250S I/O error or device error [RC=256]
- DMS251E HELP processing error code nnn-description [RC=12]
- DMS907E I/O error on file *fn ft fm* [RC=256]
- DMS1144E Implicit rollback occurred for work unit *workunitid* [RC=31]
- DMS1252T Rollback unsuccessful for file pool *filepoolid*
- DMS2521E HELPCONV cannot be performed on empty file *fn ft fm* [RC=88]

Return codes:

RC
Meaning

801
Numeric format word parameter is outside valid range.

802
Format word parameter should be a number.

803
The format word is not valid.

804
Format word parameter missing.

805
The word parameter format is not valid.

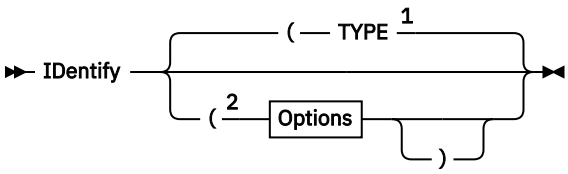
806
Undent greater than indent.

807
Excessive or negative space count generated.

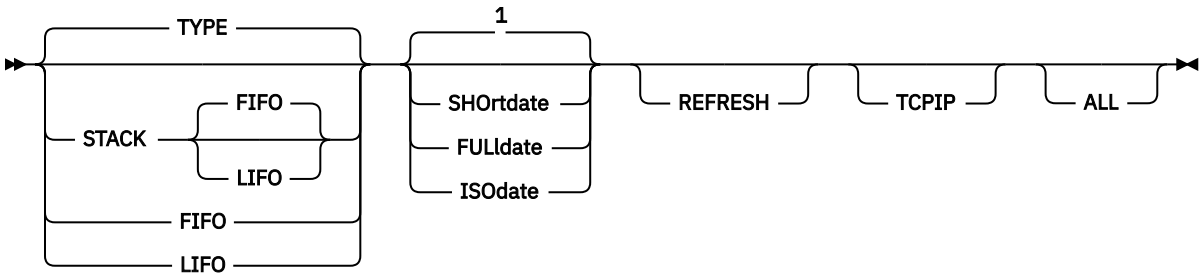
Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411
Errors in the Shared File System	“File Pool Server Messages” on page 1414
Errors in using a file	“File Error Messages” on page 1418

IDENTIFY



Options



Notes:

[1] If a date format is not specified, the user's default date format is used.
[2] You can enter Options in any order between the parentheses.

Authorization

General User

Purpose

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.

Options

TYPE
specifies the information should be displayed at the terminal. This is the default option.

STACK
specifies 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 the information should be placed in the program stack in FIFO (first in first out) order. The options STACK, STACK FIFO, and FIFO are all equivalent.

LIFO
specifies the information should be placed in the program stack in LIFO (last in first out) order. The options STACK LIFO and LIFO are equivalent.

SHOrtdate
specifies the date should be returned in *mm/dd/yy* format.

Where:

mm
specifies the month

dd
specifies the day of the month

yy
specifies the 2-digit year

FULldate
specifies the date should be returned in *mm/dd/yyyy* format.

Where:

mm
specifies the month

dd
specifies the day of the month

yyyy
specifies the 4-digit year

ISOdate
specifies the date should be returned in *yyyy-mm-dd* format.

Where:

yyyy
specifies the 4-digit year

mm
specifies the month

dd
specifies the day of the month

REFRESH
specifies the SYSTEM NETID * file will be read on this call so any new information in the file can be retrieved and saved. The default is to not reread the file unless the CPUID has changed.

TCPIP
specifies CMS is to return the TCPIP hostname for this system. This information is obtained from the TCPIP DATA file (HOSTNAME and DOMAINORIGIN entries.) If the TCPIP option is specified and:

- TCPIP DATA is not available on an accessed file space or mini-disk, error message DMS002E is returned.
- ALL option is specific concurrently, ALL is ignored.
- REFRESH are both specified, information about the user's hostname is returned and the SYSTEM NETID file is reread and that information is saved.

ALL
specifies information on all supported RSCS machines should be returned. If the processor is supporting multiple RSCS machines, there will be more than one entry in the SYSTEM NETID file. The default returns the first entry only.

Usage Notes

1. 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 retrieves the CPU serial number. (CP QUERY CPUID returns a 16-digit processor identification; however, IDENTIFY only uses digits 3 - 8.) If a SET CPUID was issued since the last IDENTIFY, and either the ALL or REFRESH option is specified, the SYSTEM NETID * file is searched for the CPUID. Otherwise, information from the previous call is used. The SYSTEM NETID * file will have one or more lines of the form:

   ```
   cpu-serial-number node rscsid
   ```

If the SYSTEM NETID file is read and 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 *.

2. The default date format used for certain CP and CMS commands can be set on a system-wide basis and also for the individual user. The system-wide default date format is set with the SYSTEM_DATEFORMAT system configuration statement. The user's default date format is set with the DATEFORMAT user directory control statement. The system-wide default and the user's default can also be set with the CP SET DATEFORMAT command. If SET DATEFORMAT is not specified for an individual user and there is no DATEFORMAT statement in the user's directory entry, the default format for that user is the system-wide default.The system-wide and user settings can be queried with the CP QUERY DATEFORMAT command. The hierarchy of possible date format settings for the IDENTIFY command, from highest priority to lowest, is:
   - IDENTIFY command option
   - User default
   - System-wide default
3. 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. For more information about SYSTEM NETID, see *z/VM: CMS Planning and Administration*.

Responses

To display the user ID information at your terminal, you would enter:

```
identify
```

This information is displayed or stacked:

```
userid AT node VIA rscsid date time zone day
```

Where:

userid
identifies the user ID of your virtual machine.

node
specifies the RSCS node of your computer. For the TCPIP option this is the hostname of your computer.

rscsid
identifies the user ID of the RSCS virtual machine. For the TCPIP option this is the TCP/IP stack virtual machine.

date
specifies the local date, in the form *mm/dd/yy*, *mm/dd/yyyy*, or *yyyy-mm-dd*.

time
specifies the local time, in the form hh:mm:ss.

zone
specifies the local time zone.

day
specifies the day of the week.

To display all the user ID information for all RSCS machines at your terminal, you would enter the following:

```
identify (type all
```

This information is displayed or stacked, one line for each RSCS machine:

```
userid AT node VIA rscsid date time zone day
userid AT node VIA rscsid date time zone day
```

```
.
.
.
```

Messages and Return Codes

- DMS002E File *fn ft* not found [RC=28]
- DMS003E Invalid option: *option* [RC=24]
- DMS056E File *fn ft [fm]* contains invalid record formats [RC=32]
- DMS070E Invalid parameter *parameter* [RC=24]
- DMS104S 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 their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

IMMCMD

Authorization

General User

Purpose

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.

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 the command has been established. A return code of 44 indicates 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 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 the Immediate command has been issued since the last invocation of IMMCMD SET or IMMCMD STATUS for that Immediate command.

name
the 1-8 character name of the Immediate command that is established (SET), canceled (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 REXX).
2. All Immediate commands established by the IMMCMD command can be explicitly canceled. If these Immediate commands are not explicitly canceled by the IMMCMD command, they are canceled automatically. They are canceled 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 established by the IMMCMD command.
3. Because no exit routine can be given control when an Immediate command (declared by 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 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 more information on the Immediate commands, see *z/VM: CMS User's Guide*.

Examples

To clear the HM immediate command, you would enter into your exec (written in REXX):

```
'IMMCMD CLEAR HM'
```

Messages and Return Codes

- DMS014E Invalid function *function* [RC=24]
- DMS047E No function specified [RC=24]
- DMS070E Invalid parameter *parameter* [RC=24]
- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS261E No immediate command name was specified [RC=24]
- DMS262E Immediate command *command* not found [RC=44]
- DMS263E Specified immediate command is a nucleus extension and cannot be cleared [RC=48]

Immediate Commands

Purpose

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 these, the:

- IMMCMD macro in an assembler language program

  For more information, see *z/VM: CMS Macros and Functions Reference*.
- IMMCMD command within an exec (CMS EXEC, EXEC 2, REXX)

  For more information, see “IMMCMD” on page 404.
- NUCXLOAD command with the IMMCMD option specified.

  For more information, see “NUCXLOAD” on page 580.

For more information on the Immediate commands, see *z/VM: CMS User's Guide*.

HB (Halt Batch Execution)

Authorization

General User

Purpose

Use the HB (Halt batch execution) command to stop the execution of a CMS batch virtual machine at the end of the current job.

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 (Halt Interpretation)

Authorization

General User

Purpose

Use the HI (Halt Interpretation) command to cause all currently executing REXX or EXEC 2 programs or macros to terminate execution without destroying the environment (as HX would).

HO (Halt Tracing)

Authorization

General User

Purpose

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.

HT (Halt Typing)

Authorization

General User

Purpose

Use the HT (Halt typing) command to suppress all terminal output generated by any CMS command or your program that is currently executing.

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 writes text to the console.
4. When using full-screen CMS, HT purges any nonpriority output in the queue for the virtual screen to which message class CMS is routed. For more information on message class routing, see "VSCREEN ROUTE" on page 1184. All priority output is displayed, such as messages and warnings through IUCV, ECHO output, and output from applications that use the LINEWRT macro coded with PRIOR=YES.

HX (Halt Execution)

Authorization

General User

Purpose

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.

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 name definitions 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, the data is not logged in a LOGFILE for the command or program that is executing. For more information, see “SET LOGFILE” on page 965.
6. If you issue HX while running a CMS multitasking-enabled or POSIX-enabled program, for example, PING, the only valid command at the VM READ is BEGIN. Any other CMS command issued will be treated as if it is the BEGIN command. You will only see the Ready; message.

RO (Resume Tracing)

Authorization

General User

Purpose

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.

RT (Resume Typing)

Authorization

General User

Purpose

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.

Usage Notes

1. Program execution continues, and displaying continues from the current point of execution in the program.
2. Any terminal output 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 (Suspend Tracing)

Authorization

General User

Purpose

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.

Usage Notes

To resume tracing, issue the RO command.

TE (Trace End)

Authorization

General User

Purpose

Use the TE (Trace End) command to stop all tracing of your REXX or EXEC 2 program or macro.

TS (Trace Start)

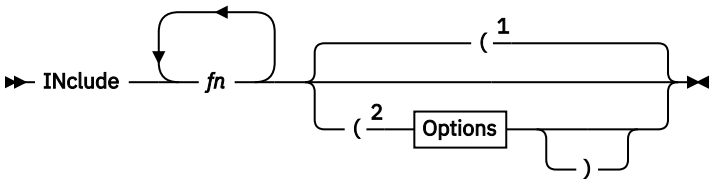
Authorization

General User

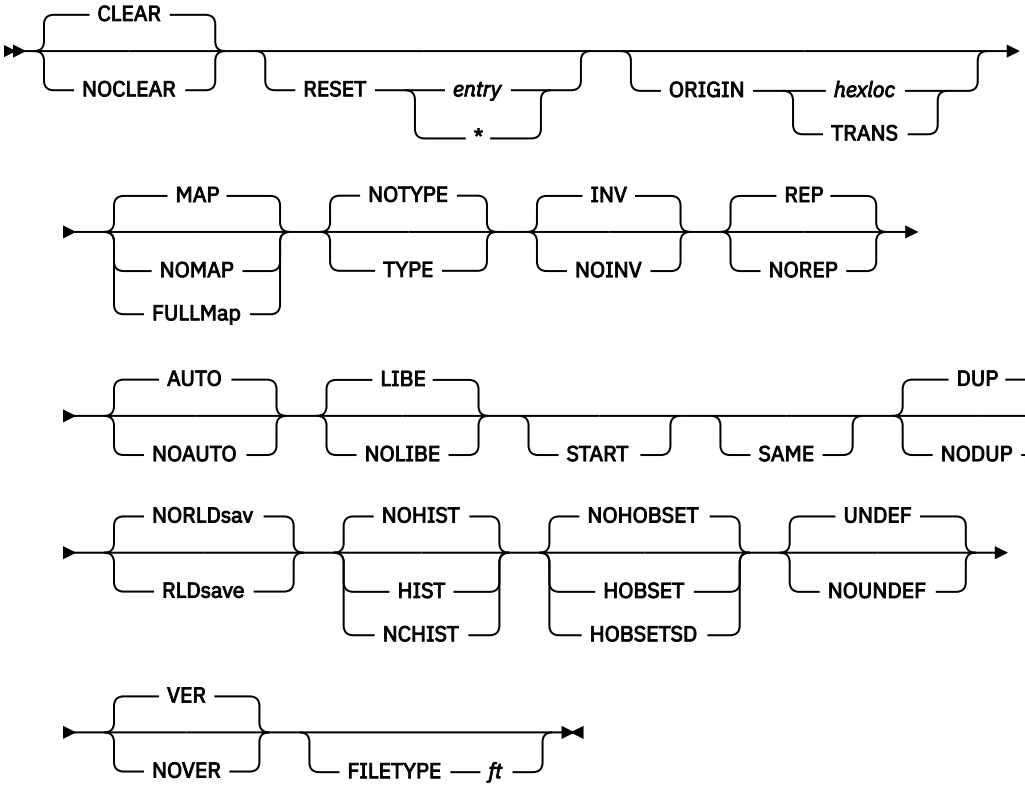
Purpose

Use the TS (Trace Start) command to start tracing of your REXX or EXEC 2 program or macro.

INCLUDE



Options



Notes:

[1] The defaults you receive appear above the line in the Options fragment.
[2] You can enter Options in any order between the parentheses.

Authorization

General User

Purpose

Use the INCLUDE command to read one or more text files (containing relocatable object code) from a minidisk or directory and 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, see “LOAD” on page 471.

Note: The SET LOADAREA command affects the outcome of the LOAD and INCLUDE commands.

Operands

fn
are the names of the files to be loaded into storage. The default file type is TEXT unless the FILETYPE option specifies the file type on the current invocation of INCLUDE or if the file type option was used on a previous invocation of LOAD/INCLUDE within the same load sequence. Files must consist of relocatable object code such as that produced by the OS language processors. If a GLOBAL TXTLIB command has identified one or more TXTLIBs, *fn* may indicate the name of a TXTLIB member.

Options

Note: Some nondefault options set on a previous LOAD or INCLUDE command retain that setting if the SAME option is used on a subsequent INCLUDE command. Some options always return to the default setting. Other options retain the previous setting until explicitly reset and are not affected by the SAME options. For more information, see Usage Note "1" on page 420.

CLEAR
clears the load area in storage to binary zeros before the files are loaded. The default is CLEAR.

NOCLEAR
does not clear the load area before loading. NOCLEAR is only valid when loading into the transient area.

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 address 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, nonrelocatable 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 on your minidisk or directory accessed as A. The default is MAP.

NOMAP
does not add information to the load map on your minidisk or directory accessed as A.

FULLMap
adds information to the variable format, full-information load map. This option is ignored if MAP was specified or defaulted on the previous LOAD or INCLUDE command.

NOTYPE
does not display the load map at the terminal. NOTYPE is the default.

TYPE
displays the load map at your terminal.

INV
writes card images that are not valid in the load map file. The default is INV.

NOINV
does not write the card images, not valid in the load map file.

REP
writes Replace (REP) statement images in the load map file (unless the NOMAP option is specified). For information on the Replace (REP) statement, see "LOAD" on page 471. The default is REP.

NOREP
suppresses the writing of Replace (REP) statements in the load map file.

AUTO
searches your disks for text files to resolve undefined references. The default is AUTO.

NOAUTO
suppresses automatic searching for text files.

LIBE
searches the text libraries defined by a previously issued GLOBAL command for unresolved references. The default is LIBE.

NOLIBE
does not search any text libraries for unresolved references.

START
begins execution after loading is completed.

SAME
retains certain nondefault options NOMAP, TYPE, NOINV, NOREP, NOAUTO, NOLIBE, and NOVER are used in a previous LOAD or INCLUDE command. For more information, see Usage Note "1" on page 420.

DUP
displays warning messages at your virtual console when a duplicate CSECT is encountered during processing. The duplicate CSECT is not loaded. The default is DUP.

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. The default is NORLDsav.

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.

UNDEF
displays undefined entry point names at the virtual console and prints them in the load map file (unless the NOMAP option is specified). The default is UNDEF.

NOUNDEF
suppresses the display and printing of undefined entry point names.

VER
writes Verify (VER) statement images in the load map file (unless the NOMAP option was specified). For information on the Verify (VER) statement, see "LOAD" on page 471. The default is VER.

NOVER
suppresses the writing of Verify (VER) statements in the load map file.

FILETYPE *ft*
specifies the file type of the file(s) to be searched for and loaded. This file type is not used during text library searches (LIBE option). If the FILETYPE option was specified on a previous LOAD or INCLUDE within the same load sequence, the file type specified remains in effect until it is specifically reset using the FILETYPE option.

NOHIST
does not save history information from text files. The default is NOHIST.

HIST
saves the history information 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 previously requested from other text files. (For example: The HIST option was specified on the LOAD command and other INCLUDE commands).

NCHIST
retains only uncommented history records from text files for inclusion in the module later created by GENMOD. Commented history records (that is, those with an asterisk (*) in column one) will be ignored.

NOHOBSET
specifies the high-order bit of all V-type constants (VCONs) will be left unchanged. This is the default.

HOBSET
specifies the high-order bit is to be turned on for V-type constants (VCONs) of SD (CSECTs) and LD (ENTRYs) types that have AMODE ANY/31 addresses.

HOBSETSD
specifies the high-order bit is to be turned on for V-type constants (VCONs) of SD (CSECTs) type that have AMODE ANY/31 addresses.

Usage Notes

1. If any of the following nondefault options were set on a previous LOAD or INCLUDE command, you can retain these options on a subsequent INCLUDE command by using the SAME option:

   NOMAP, TYPE, NOINV, NOREP, NOAUTO, NOLIBE, NOVER

   When using the SAME option, if you want to return one of these options to its default setting, you must explicitly reset it on the INCLUDE command. You must specify SAME on each invocation of INCLUDE for the options effected by SAME to remain in effect.

   If any of the following nondefault options were set on a previous LOAD or INCLUDE command, they retain that setting until specifically reset on a subsequent INCLUDE command, and are not affected by the SAME option:

   FULLMap, RLDSAVE, RMODE, HIST, NCHIST, FILETYPE, NOUNDEF

   (The MAP option, which is a default option, is also retained.)

   If one of the following nondefault options was set on a previous LOAD or INCLUDE command, it automatically returns to its default setting on a subsequent INCLUDE command unless you specifically reset it:

   NOCLEAR, NODUP, HOBSET, HOBSETSD

   Note: The default in the case of NOHOBSET/HOBSET/HOBSETSD is determined by the default value specified on the DEFAULTS command. For more information, see Usage Note “10” on page 421.

   For example, if you enter the command

```
load myprog (noclear nomap norep noauto nolibe noundef nover
nchist)
```

the file named MYPROG TEXT is brought into real storage. Including the defaults, the following LOAD options are in effect:

NOCLEAR, NOMAP, NOTYPE, INV, NOREP, NOAUTO, NOLIBE, DUP, NORLD, NOPRES, NOUNDEF, NOVER, NCHIST

If you then enter the command

```
include mysub (map undef same)
```

the file named MYSUB TEXT is appended to MYPROG TEXT. The following LOAD/INCLUDE options are in effect:

CLEAR, MAP, NOTYPE, INV, NOREP, NOAUTO, NOLIBE, DUP, NORLD, NOPRES, UNDEF, NOVER, NCHIST

2. When the INCLUDE command is issued, the loader tables are not reset.
3. An INCLUDE that intersects a previously loaded nonrelocatable program causes that program to be deleted from storage.
4. Attempting to do an INCLUDE after a LOADMOD of a relocatable module may have unpredictable results.
5. Because of the interactive environment of CMS, in a 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 file 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 file ESD, loading will restart below the 16MB line. Because other commands may be executed between INCLUDE commands, the disks or directories used to load the TEXT files specified on the LOAD or previous INCLUDE commands may be released or changed. The CMS LOAD process will restart in the environment that is in effect at the point of the restart.
6. 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.
7. 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.
8. If the NOUNDEF option was used on the LOAD command, the UNDEF option can be used on the last INCLUDE command to display any entry points that are still unresolved. If any are found, the return code from INCLUDE is 4; if none are found, the return code is 0 (assuming no other errors).
9. When using the FILETYPE option to specify the text file type, you should also use the NOAUTO option to suppress the automatic searching for text files, unless *all* the files to be searched for and loaded have the same file type.
10. You can use the DEFAULTS command to specify the MAP|FULLMap|NOMAP option, or the NOHOBSET|HOBSET|HOBSETSD option as the default for the INCLUDE command, or override the default option. However, the option you specify when entering the INCLUDE command overrides the one specified in the DEFAULTS command. This allows you to customize the default for the INCLUDE command, yet override it when you desire. For more information, see “DEFAULTS” on page 153.
11. When the FULLMap option is specified, the load map created from a sequence of LOAD and INCLUDE commands will be named "*fn* LOADMAP A5", where *fn* is the first file name specified on the LOAD. If NOMAP is specified on the LOAD command, *fn* is the first file name specified on the first successful INCLUDE with the FULLMap option. If the MAP option is specified, the load map will be named "LOAD MAP A5". If a file already exists with that name, it will be replaced. The format of the load map file will not change once generated by using LOAD or INCLUDE in the same load sequence. For information on the load map produced with the MAP and FULLMap options, see “LOAD” on page 471.

Responses

```
DMS740I  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 a not valid card has been found. The message and the contents of the not valid card (xxx...xxx) are listed in the load map file. The card that is not valid is ignored and loading continues.

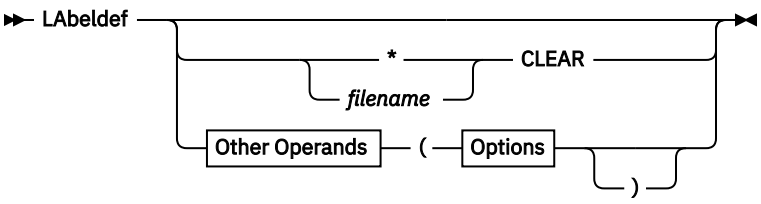
Messages and Return Codes

- DMS001E No filename specified [RC=24]
- DMS002E File[(s)] *fn* TXTLIB not found [RC=28]
- DMS003E Invalid option: *option* [RC=24]
- DMS005E No *option* specified [RC=24]
- DMS021E Entry point *name* not found [RC=40]
- DMS029E Invalid parameter *parameter* in the option *option* field [RC=24]
- DMS055E No entry point defined [RC=40]
- DMS056E File *fn ft [fm]* contains invalid [RLD] record formats [in *entryname*] [RC=32]
- DMS099E CMS/DOS environment not active [RC=40]
- DMS104S Error *nn* reading file *fn ft fm* from disk [RC=31 | 55 | 70 | 76 | 99 | 100]
- DMS105S Error *nn* writing file *fn ft fm* on disk [RC=31 | 55 | 70 | 76 | 99 | 100]
- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS116S Loader table overflow [RC=104]
- DMS168S Pseudo register table overflow [RC=104]
- DMS169S ESDID table overflow [RC=104]
- DMS201W The following names are undefined: *namelist* [RC=4]
- DMS202W Duplicate identifier *identifier* [RC=4]
- DMS203W SET LOCATION COUNTER *name* undefined [RC=4]
- DMS206W Pseudo register alignment error [RC=4]
- DMS257T Internal system error at address *addr* (offset *offset*)
- DMS379E INCLUDE address *at or above*|*below* 16MB conflicts with LOAD address *below*|*at or above* 16MB, INCLUDE failed [RC=88]
- DMS380E Storage at origin *addr* in use, *file* not loaded. [RC=104]
- DMS381E Insufficient storage available below 16MB to load *file*. [RC=88]
- DMS623S Module cannot be loaded at location *location*—this area is available for system use only [RC=88]
- DMS625S There are too many items that require relocation to save all of the RLD information [RC=104]
- DMS907T I/O error on file *fn ft fm*
- DMS994W Restrictive RMODE encountered in CSECT *cname*, LOAD continues below 16MB.
- DMS997E The specified ORIGIN address is outside the virtual machine size, LOAD|INCLUDE failed. [RC=64]
- DMS1144E Implicit rollback occurred for work unit *workunitid* [RC=31]
- DMS1252T Rollback unsuccessful for file pool *filepoolid*

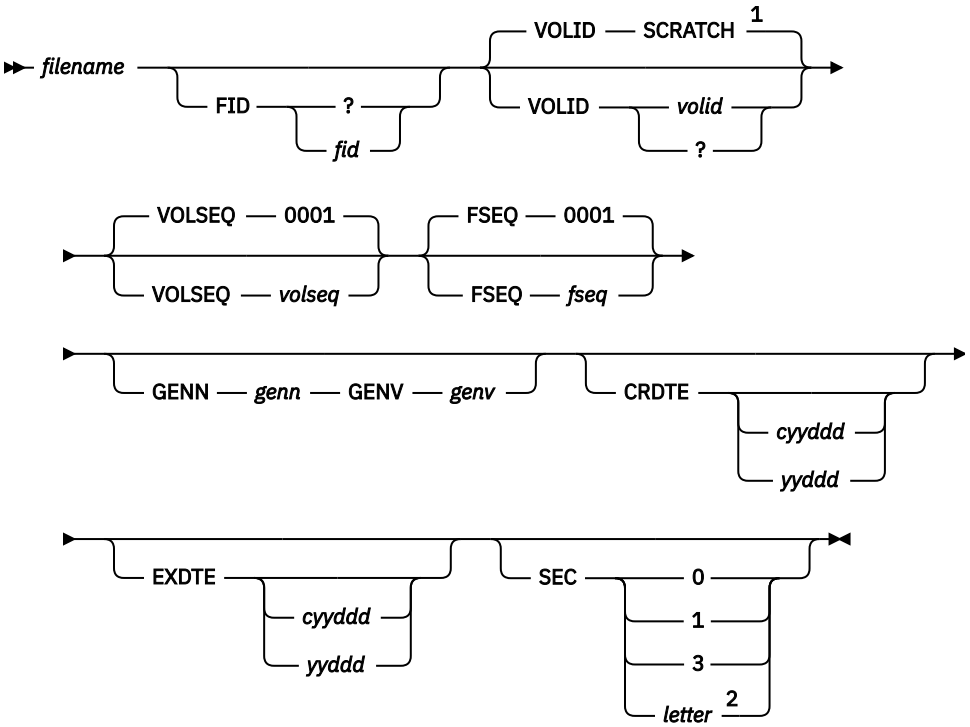
Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in the Shared File System	“File Pool Server Messages” on page 1414
Errors in using a file	“File Error Messages” on page 1418

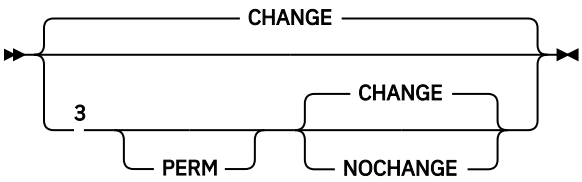
LABELDEF



Other Operands



Options



Notes:

[1] SCRATCH is the default value for OS simulation. CMS001 is the default value for CMS/DOS.
[2] Any "letter" A through Z may be specified, but only for AL and AUL tapes. A through Z will be accepted when a prior FILEDEF has been issued for this "*filename*" with the label type of AL or AUL.
[3] You can enter Options in any order between the parentheses.

Authorization

General User

Purpose

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. For more information on the TAPESL macro, see *z/VM: CMS Macros and Functions Reference*.

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 FT*nn*F001.

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, this 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 eight 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 eight 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. See Usage Note “9” on page 428.

VOLID *volid*
VOLID ?
VOLID SCRATCH
supplies the volume serial number (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 *volid*). The valid *volid*'s can be uppercase alphabetic, numeric, or these characters, for the:

- AL tapes:

```
" ; : < = > ? - . + / , * & % ! ( ) '
```

- SL tapes:

```
¢ $ ǂ - / @
```

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. This is the default.

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). GENN is a matched entry with GENV to specify as OS Generation Data Group (GDG) number; '.G*nnnn*V*nn*'. If one field is entered, the other field must also be entered.

GENV *genv*
is the generation version (1-2 numeric characters). GENV is a matched entry with GENN to specify as OS Generation Data Group (GDG) number; '.G*nnnn*V*nn*'. If one field is entered, the other field must also be entered.

CRDTE *cyyddd* or *yyddd*
is the Julian creation date.

EXDTE *cyyddd* or *yyddd*
is the Julian expiration date.

Note: For CRDTE and EXDTE, the Julian date can be either 5 or 6 digits. The *c* in the 6 digit format indicates the century, as: Where:

9
specifies the 1900's

0
specifies the 2000's

1
specifies the 2100's

If the date is entered in the 5 digit format, the century will default based on the 2 digit year number. If *yy* is greater than 50, the century will default to the 1900's. Therefore the date will be shown as 9*yyddd* in all QUERY LABELDEF responses. If it is equal to or less than 50, it will default to the 2000's.

SEC
specifies security classification (0, 1, 3, or A-Z). For more information on the meaning of security classification on tape files, see *Magnetic Tape Labels and File Structure for MVS/DFP*.

Note: This number has no effect on how the file is processed. If you are writing to a tape, the SEC value is written to the tape label. If you are reading a tape, CMS will compare the SEC value specified in the LABELDEF command to the SEC value written in the label; if the two values are equal, you are allowed access.

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. This is the default.

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 *labeldefid* specified in the LABID parameter.

VOLID
is CMS001 for CMS/DOS.

For OS simulation, the actual VOLID from the tape mounted is used if processing an "SL/SUL" or "AL/AUL" tape file.

In multivolume processing, for subsequent volumes, the actual *volid* from the tape mounted is used from the Volume Serial Number field in the volume label (VOL1) to record the multivolume order. If no explicit LABELDEF was issued, each new tape mounted has its serial number recorded as it is read. For OUTPUT operations, the *volid* of the first volume in the set is written to the 'Data Set Serial Number' field in the HDR1 label, to identify the logical start to the data.

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 for IBM standard labeled tapes and blank for ANSI labeled tapes. (If a 0 is specified for an ANSI labeled tape, the SEC code will be set to blank, however, a blank cannot be specified with the SEC option.)

Usage Notes

1. To check a field in an input label, specify it on your LABELDEF command for the label. (Even if you do not specify a value for the file identifier field, it is checked to make sure it is not zeros or blanks.) 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 99285
   ```

   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 these 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 these values:

   ```
   file ID                 MASTER
   file sequence number    0002
   volume sequence number  0001
   creation date           date when label is written
   expiration date         99285
   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, all the volume IDs and SCRATCH will be displayed. If SCRATCH was entered at command time and the file has been opened, 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 99300
   filedef filez tap1 sl volid vol4
   ```

   define filez as a labeled tape file on tape 181. The volume serial is VOL4, the file identifier is PAYROLL, and the file sequence number is 0002. Expiration date is day 300 in 1999. 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:

   ```
   TAPESL HOUT,181,LABID=FILE14
   ```

   A CMS TAPEMAC command could use the same LABELDEF as:

   ```
   tapemac maclib sl file14
   ```

   In all the preceding examples, the LABELDEF statement must be issued before the program or command is executed.
5. The LABELDEF command does not support multivolume tape processing for the 9346 cartridge tape drive.
6. In OS simulation, multivolume tape processing only applies to CMS OS QSAM simulation for IBM standard labeled (SL) and ANSI labeled (AL) tapes.

   In CMS/DOS simulation, multivolume tape processing is not supported. The multi or alternate volume parameters of FILEDEF or LABELDEF are not applicable.

   In a CMS/DOS environment, it is not possible to create or process multivolume tapes. If a large file needs to be copied to tapes the COUNT and SKIP parameters of the IDCAMS REPRO function can be used to create multiple tapes where each tape contains only a segment of the file.
7. If you want to specify an ANSI-specific parameter in your LABELDEF command, you must first issue a FILEDEF command with the same ddname, and then specify AL or AUL labels. Examples of ANSI-specific parameters are:
   - SEC codes A-Z
   - FID with ANSI special characters
   - VOLID with ANSI special characters
8. CMS Standard Label tape processing will continue to support the existing customer convention of using either a 99365 or 99366 EXDTE entry to indicate a tape label date that will *never expire*. Tapes mounted containing a HDR1 label with one of these expiration dates will receive an UNEXPIRED FILE warning which the user must respond to with either an IGNORE (write to the tape) or an ERROR (don't write to the tape).
9. If the user specifies the GENN and GENV parameters on the LABELDEF, the command will automatically append a Generation Dataset Group (GDG) number in the form: '.G*nnnn*V*nn*' to the user's 17–character HDR1 File Identifier (FID) for OS compatibility. This automatic append of the

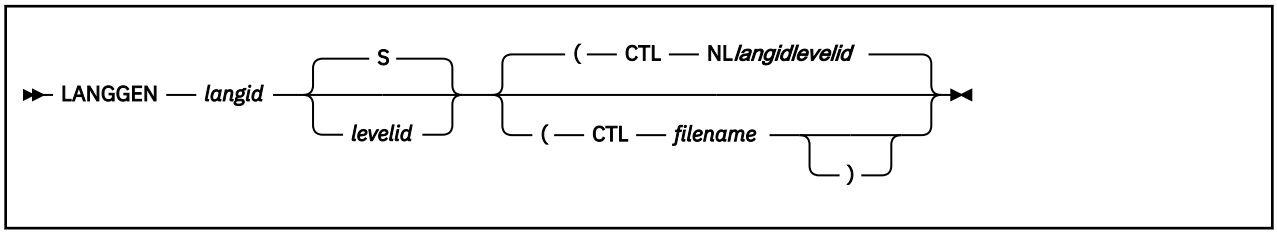
GDG number will cause left-ended truncation of the original FID data if the FID data is longer than 8 characters. Note, for visibility the LABELDEF list function will force the display of any FID modified because of the use of the GENN and GENV GDG parameters.

10. For more information on the CMS tape label processing, see *z/VM: CMS Application Development Guide for Assembler*.

Messages and Return Codes

- DMS003E Invalid option: *option* [RC=24]
- DMS029E Invalid parameter *parameter* in the option *option* field [RC=24]
- DMS050E Required matching parameter missing; GENN GENV [RC=24]
- DMS065E *option* option specified twice [RC=24]
- DMS066E *option1* and *option2* are conflicting options [RC=24]
- DMS070E Invalid parameter *parameter* [RC=24]
- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS220R Enter data set name:
- DMS221E Invalid data set name [RC=24]
- DMS324I No user defined LABELDEFs in effect
- DMS438E Volid *volid* is a duplicate entry [RC=24]
- DMS439E Volid *volid* is an invalid entry [RC=24]
- DMS441R Enter volid information
- DMS442E SCRATCH may only be used as the last volid for the file [RC=24]
- DMS649E Extraneous parameter(s) [RC=24]
- DMS704I Invalid CLEAR request

LANGGEN



Authorization

Saved Segment Administrator

Purpose

Use the LANGGEN command to get all the text files created by the LANGMERG command for a language and save them in a saved segment. LANGGEN also saves the CP message repository on DASD.

You should use LANGGEN for these purposes:

- Adding a language to your system
- Adding an application to a language saved segment
- Regenerating a language saved segment after a language-related file has been changed

Note: If you are working with logical saved segments containing applications, you should be using the SEGGEN command rather than the LANGGEN command. In that case, the LANGGEN command should be used only to save the CP message repository on DASD. For more information on physical or logical segments, see *z/VM: Saved Segments Planning and Administration*.

Operands

langid
is the language ID of the language whose files you are saving in a saved segment. A language ID may be 1-5 characters in length and must be made up of only CMS file system characters.

levelid
is a one-character (A-Z,0-9) level ID that identifies the level or version of the saved segment being built. It is used as the third character of the saved segment name after NL. If omitted, it defaults to S.

CTL *filename*
specifies the file name of a control file that identifies the language files to be saved. The default control file name is NL*levelidlangid*. The file type of this control file must be LANGGCTL. The file can reside either on a minidisk or in an SFS directory. The contents of the control file are described below.

Usage Notes

1. The saved segment created by LANGGEN is named NL*levelidlangid*.
2. You must have the system national language set on your virtual machine to use the LANGGEN command. The virtual machine must be large enough to include the segment being saved.
3. You can use the *levelid* operand of LANGGEN when you want to have more than one version of a language saved segment. For instance, you may have a uppercase English saved segment called NLSUCENG that contains everything you want. However, you want to add some things to the saved segment while preserving your original copy. You could call this new uppercase English saved segment NL2UCENG.

   Note: The saved segment whose name contains the new level ID must be defined using the CP DEFSEG and CP SAVESEG commands. This new language saved segment can be used by building

a new CMS nucleus and specifying the new level ID in response to message 295R during the build process. You must have a saved segment with this level ID for every language that will be used with the new CMS nucleus.

4. When it successfully saves language information, LANGGEN tries to add an entry with that language ID to the SYSTEM LANGUAGE file. For this to be successful, a copy of the SYSTEM LANGUAGE file must be on a read/write file mode accessed before any read-only file mode that contains a SYSTEM LANGUAGE file. To ensure this, copy the SYSTEM LANGUAGE file from the S-disk to file mode A before running LANGGEN. When you are done, the SYSTEM LANGUAGE file on file mode A must be copied back to the system disk. If you do not have read/write access to the system disk, ask someone who does to copy it back to the system disk for you.
5. LANGGEN creates a map on file mode A that shows the location of each application text file that was loaded into the saved segment. The file ID of this map is NL*levelidlangid* DCSSMAP.
6. When building a language segment, the corresponding storage for the segment must be available. If the storage is not available, these messages are issued:

   ```
   DMS343E Storage in range addr1-addr2 for segname in use.
   DMS283E The segname saved segment could not be reserved; return
             code 41 from SEGMENT RESERVE.
   ```

   To reserve the storage, issue the SEGMENT RESERVE command for this segment immediately following an IPL and the SEGMENT RELEASE command just before issuing LANGGEN.

   For some segments located at high addresses in your virtual machine, issuing the SEGMENT RESERVE and SEGMENT RELEASE commands may not solve this problem. LANGGEN may be using storage from this area, thus making it unavailable for loading segment information. In this case, you must either move the segment to a lower address or build the segment in a logical segment using the SEGGEN command.

For more information about saved segments, see *z/VM: Saved Segments Planning and Administration*.

LANGGEN's Control File

You must create a LANGGEN control file to issue the LANGGEN command.

For CP, the LANGGEN control file contains the identifier of the CP message repository. The repository has a file name of HCPMES*cc*.

Where:

cc
specifies the country code of the language

LANGGEN loads this message repository on the DASD that was previously defined with a DEFSEG command. LANGGEN then saves the repository. If you are using logical saved segments, the LANGGEN control file contains only a CP record for the CP message repository.

For CMS and other applications, this control file identifies the language text files that LANGGEN loads into a saved segment for the language. These language text files, which are produced by the LANGMERG command, have a file ID of *applid*NLS TXT*langid* (although an application could supply merged text files with a file type other than TXT*langid*).

The LANGGEN control file may contain the following types of records:

1. Comment:

   ```
   *comment
   ```

   Anything may follow the * on a comment record. The asterisk must be the first character on the record.
2. Records that identify the file IDs of language files:

   ```
   fn [ft fm] [(VMCTL cntrlfn[)]]
   ```

The *fn* variable is the file name (the file type and file mode are optional) of the language file where the information is to be read. The file names should be in this format:

HCPMES*cc*
for a CP message repository

DMSNLS
for a CMS language file

***applid*NLS**
for an application language file

Note: You do not need to specify the country code (*cc*) on the file name of the CP message repository if you omit the file type and you specify the VMCTL *cntrlfn* option.

The VMCTL *cntrlfn* option specifies a control file LANGGEN should use to determine the file ID of the language file (object deck):

- If you specify a file type on the language file record, LANGGEN uses the file ID as specified, regardless of whether you specify the VMCTL *cntrlfn* option in the LANGGCTL file.
- If you do not specify a file type on the language file record, but you do specify the VMCTL *cntrlfn* option on either that record or a previous language file record in the LANGGCTL file:
  a. LANGGEN looks up the language identifier (*langid*) in the VMFNLS LANGLIST file. If VMFNLS LANGLIST is not found, message 2 (`File VMFNLS LANGLIST * not found`) return code 28 is issued.
  b. If the file name from the language file record in the LANGGCTL file is six or fewer characters long, LANGGEN appends the two-character country code for that language from VMFNLS LANGLIST to the end of the language file name; otherwise, the file name is used as is.
  c. LANGGEN calls VMFLDS to determine the object deck file type.
  d. If a control file name is not specified after the VMCTL keyword in the LANGGCTL file, LANGGEN displays message 387 format 5 (`Missing file name for VMCTL operand`) on the screen for each language file name listed with or under that keyword (until the end of the LANGGCTL file or a valid VMCTL *cntrlfn* option is reached).
  e. If the control file cannot be found, message 2 (`File` *`fn ft fm`* `not found`) is issued.
- If you do not specify a file type on the language file record and you do not specify the VMCTL *cntrlfn* option on either that record or a previous language file record in the LANGGCTL file, LANGGEN uses the file name from the language file record and assumes the file type is TXT*langid*.

General Usage Notes

1. Each control file name can be different.
2. You only need to specify the VMCTL *cntrlfn* option on the first language file record in the LANGGCTL file.
3. The VMCTL *cntrlfn* option applies to the file ID it is specified with and all subsequent file IDs until either:
   - Another VMCTL *cntrlfn* option is encountered
   - End of file (EOF) is reached
4. The merged CMS object output from LANGMERG continues to have a file identifier of DMSNLS TXT*langid*. This file is the latest level of the merged DMS object decks. This means all LANGGEN file type searching need only apply to the CP message repository file and other application language files listed in the LANGGCTL file.
5. The order of the CMS and CMS application files in the LANGGEN control file determines their order in the saved segment. Also, you cannot have a LANGGEN control file that identifies only an application; you must include an entry for CMS or else the language is not saved.

Examples

Example 1

Here is a sample LANGGEN control file for uppercase English (*langid*=UCENG). The file identifies text files for CP, CMS, and two applications named AP1 and AP2.

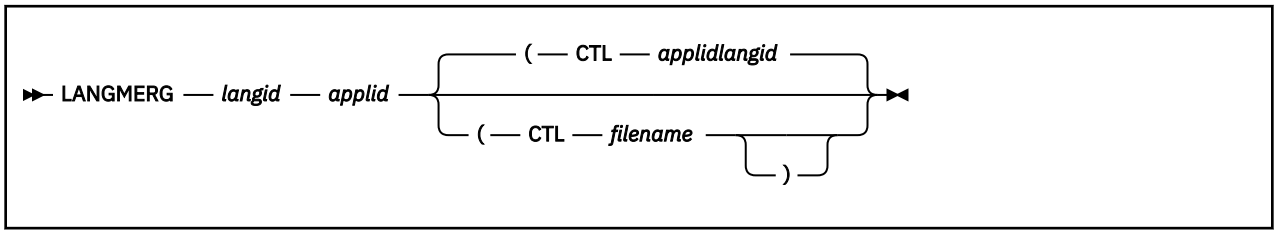
```
HCPMESE TEXT    F
DMSNLS TXTUCENG   F
AP1NLS TXTUCENG   G
AP2NLS TXTUCENG   B
```

When adding a new file for an application, you must be sure to include that application's text file in the LANGGEN control file.

Messages and Return Codes

- DMS002E File *fn ft fm* not found [RC=28]
- DMS006E No read/write disk accessed [RC=36]
- DMS032E Invalid filetype *ft*
- DMS048E Invalid filemode *mode*
- DMS069E Disk *mode* not accessed
- DMS104S Error *nn* reading file *fn ft fm* from disk [RC=EXECIO rc]
- DMS105S Error *nn* writing file *fn ft fm* [RC=EXECIO rc]
- DMS210E File *fn ft* is on a read/only disk [RC=28]
- DMS235E Error *n* in input text file *fn ft fm* [RC=10]
- DMS252E Invalid filename *fn* [RC=20]
- DMS276E Invalid language ID *langid* [RC=24]
- DMS279E Application DMS not found in the language saved segment [RC=32]
- DMS282E Error(s) occurred while creating *fn ft fm*; check *fn ft fm* for details [RC=32]
- DMS283E The *name* saved segment could not be reserved; return code *rc* from SEGMENT RESERVE [RC=128]
- DMS284E The saved segment is not completely inside the virtual machine [RC=88]
- DMS285I CP repository saved [RC=0]
- DMS285E CP repository not saved [RC=104]
- DMS286E The saved segment is too small for the data being stored [RC=40]
- DMS287E You must have a special privilege class to successfully issue the LANGGEN command [RC=40]
- DMS288I *ssname* saved segment not saved [RC=40]
- DMS289E The default language, *langid*, must be active [RC=104]
- DMS290E Duplication applications specified in control file *fn ft fm* [RC=32]
- DMS291E Error occurred while loading the saved segment [RC=104]
- DMS292W Text data will be loaded at X'20000' in user area; user data may be overwritten
- DMS294E Invalid language level ID *levelid* [RC=24]
- DMS312W Language not generated - no text decks specified in control file *fn ft fm* [RC=4]
- DMS346E Error *nn* loading *fn ft* from disk [RC=32]
- DMS387E Missing *valuetype* for *operand* operand [RC=24]

LANGMERG



Authorization

Saved Segment Administrator

Purpose

Use the LANGMERG command to combine all the language-related files for an application into one text file. You can then use the LANGGEN command to load this single text file into a physical saved segment as a language segment, or you can use the SEGGEN command to load this text file into a logical saved segment.

You should use the LANGMERG command for these purposes:

- When you have changed something in a language file and you want to recombine all the updated language information for an application.
- When you are building a language text file for an application.

Operands

langid
is the language ID of the language whose text file is being created. A language ID may be 1-5 characters in length and must be made up of only CMS file system characters.

applid
is the application ID of the application whose text file is being created. This must be three characters long.

CTL *filename*
specifies the file name of a control file that identifies the language files to be combined. The default control file name is *applidlangid*. The file type of the control file must be LANGMCTL. The file can reside either on a minidisk or in an SFS directory. The contents of the control file are described below.

Usage Notes

1. The single text file created by LANGMERG has a file ID of *applid*NLS TXT*langid*.
2. When you want to use a different language file, you must either:
   - Edit the default LANGMERG control file.
   - Make your own control file and specify it as an option when you invoke LANGMERG.
3. LANGMERG creates a map on file mode A that shows where language information is stored within a text file. The file ID of this map is *applidlangid* LANGMAP.
4. The text files produced by GENCMD may be empty. LANGMERG detects this condition and notes it in the LANGMAP file. This general condition indicates no entries were produced for the file.

LANGMERG's Control File

You must create a LANGMERG control file to issue the LANGMERG command. LANGMERG's control file supplies information about the language and identifies the files to be loaded.

The LANGMERG control file may contain the following types of records:

1. Comment:

   ```
   *comment
   ```

   Anything may follow the * on a comment record. The asterisk must be the first character on the record.

2. A DISK record, which identifies the address of a disk associated with the language:

   ```
   DISK vdev
   ```

   For CMS, the DISK record identifies the HELP disk; other applications can use this record for different purposes.

3. An ETMODE record, which identifies whether the language named is a double-byte character set (DBCS) language:

   ```
   ETMODE ON|OFF
   ```

   ETMODE should be ON if the language is a DBCS language. If this record is omitted in the control file, OFF is assumed.

4. Records that identify the file IDs of language files in the language control block:

   ```
   keyword fn [ft fm] [(VMCTL cntrlfn[)]]
   ```

   keyword can be:

   MESSAGE
   for a system or user message repository

   PARSERS
   for system or user command syntax definition files

   SYNONYMS
   for a system or user national language synonym and translation file

   TRTABLES
   for translate tables

   USER
   for your installation's use

   You can specify the MESSAGE, PARSERS, SYNONYMS, and TRTABLES records only once in the control file.

   The *fn* variable is the file name (the file type and file mode are optional) of the language file where the information is to be read. The file names should be in this format:

   applid*MES*cc
   for a system message repository

   applid*UME*cc
   for a user message repository

   applid*SPA*cc
   for system command syntax definition files

   applid*UPA*cc
   for user command syntax definition files

   applid*SSY*cc
   for a national language system synonym and translation file

   applid*USY*cc
   for a national language user synonym and translation file

   applid*TRT*cc
   for translate tables

applid*USE*cc
for your installation's use

Note: You do not need to specify the country code (*cc*) on the file name if you omit the file type and you specify the VMCTL *cntrlfn* option.

The VMCTL *cntrlfn* option specifies a control file LANGMERG should use to determine the file ID of the language file (object deck):

- If you specify a file type on the language file record, LANGMERG uses the file ID as specified, regardless of whether you specify the VMCTL *cntrlfn* option in the LANGMCTL file.
- If you do not specify a file type on the language file record, but you do specify the VMCTL *cntrlfn* option on either that record or a previous language file record in the LANGMCTL file:
  a. LANGMERG looks up the language identifier (*langid*) in the VMFNLS LANGLIST file. If VMFNLS LANGLIST is not found, message 2 (`File VMFNLS LANGLIST * not found`) return code 28 is issued.
  b. If the file name from the language file record in the LANGMCTL file is six or fewer characters long, LANGMERG appends the two-character country code for that language from VMFNLS LANGLIST to the end of the language file name; otherwise, the file name is used as is.
  c. LANGMERG calls VMFLDS to determine the object deck file type.
  d. If you do not specify a control file name after the VMCTL keyword in the LANGMCTL file, LANGMERG writes message 387 format 5 (`Missing file name for VMCTL operand`) to the *ay* CMSUT2 file for each language file name listed with or under that keyword (until the end of the LANGMCTL file or a valid VMCTL *cntrlfn* option is reached).
  e. If the control file cannot be found, message 2 (`File` *`fn ft fm`* `not found`) is issued.
- If you do not specify a file type on the language file record and you do not specify the VMCTL *cntrlfn* option on either that record or a previous language file record in the LANGMCTL file, LANGMERG uses the file name from the language file record and assumes the file type is TXT*langid*.

General Usage Notes

1. Each control file name can be different.
2. You only need to specify VMCTL on the first language file record in the LANGMCTL file.
3. The VMCTL *cntrlfn* option applies to the file ID it is specified with and all subsequent file IDs until either:
   - Another VMCTL *cntrlfn* option is encountered
   - End of file (EOF) is reached
4. The merged CMS object output from LANGMERG continues to have a file identifier of DMSNLS TXT*langid*. This file is the latest level of the merged DMS object decks.
5. The order of the records in the LANGMERG control file determines where the application's files will go in the saved segment. You may be able to improve your system's performance by changing the order of these files.

Examples

Use the following LANGMERG control file for CMS in uppercase English (application ID for CMS is DMS; language ID for uppercase English is UCENG):

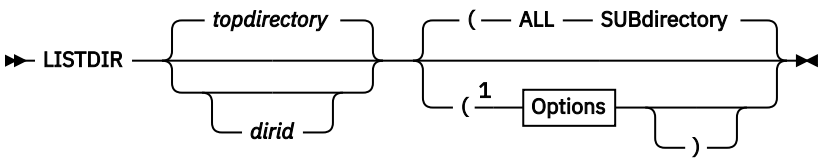
```
*
* This is the LANGMERG control file for uppercase English in CMS.
* The file ID of this control file is DMSUCENG LANGMCTL.
*
DISK 19D
ETMODE OFF
MESSAGE  DMSMES (VMCTL DMSVM
PARSERS  DMSSPA
SYNONYMS DMSSSY
```

```
TRTABLES DMSTRT
USER     DMSUSE
```

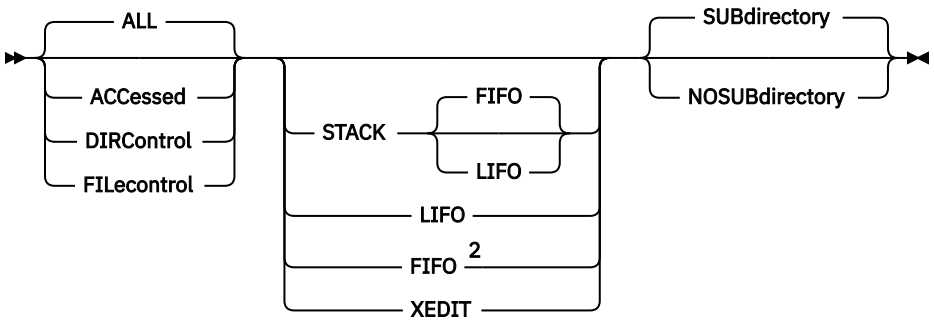
Messages and Return Codes

- DMS002E File *fn ft fm* not found [RC=28]
- DMS006E No read/write disk accessed [RC=36]
- DMS104S Error *nn* reading file *fn ft fm* from disk [RC=*rc* from EXECIO]
- DMS105S Error *nn* writing file *fn ft fm* on disk [RC=*rc* from EXECIO]
- DMS252E Invalid filename *fn* [RC=20]
- DMS276E Invalid language ID *langid* [RC=24]
- DMS282E Error(s) occurred while creating *fn ft fm*; check *fn ft fm* for details [RC=32]
- DMS440W Merged text deck not created - no text decks were specified in control file *fn ft fm* [RC=4]
- DMS770E Invalid application ID *applid* [RC=24]

LISTDIR



Options



Notes:

[1] You can enter Options in any order between the parentheses.
[2] XEDIT option is valid only from XEDIT.

Authorization

General User

Purpose

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

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. You can also use a file mode for the *dirid* if the directory is already accessed. If *dirid* is not specified, the list begins with your top directory. For more information, see the description of the ACCessed option.

For more information on *dirid*, see "Naming Shared File System (SFS) Directories" on page 6.

Options

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

ACCessed
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.

DIRControl
lists only directories with the directory control (DIRCONTROL) attribute. Directories are listed even if they are not accessed.

FILecontrol
lists only directories with the file control (FILECONTROL) attribute. Directories are listed even if they are not accessed.

SUBdirectory
lists the directory specified by *dirid* and its subdirectories. The specified directory can not be open when using this operand. The default is SUBDIRECTORY.

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 accessed in the CMS search order is listed.

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

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

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

XEDIT
places the list in the file 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 REXX exec with ADDRESS COMMAND in effect.

Examples

User SMITH enters,

```
listdir
```

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

```
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
```

For more information on using LISTDIR, see *z/VM: CMS User's Guide*.

Responses

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.

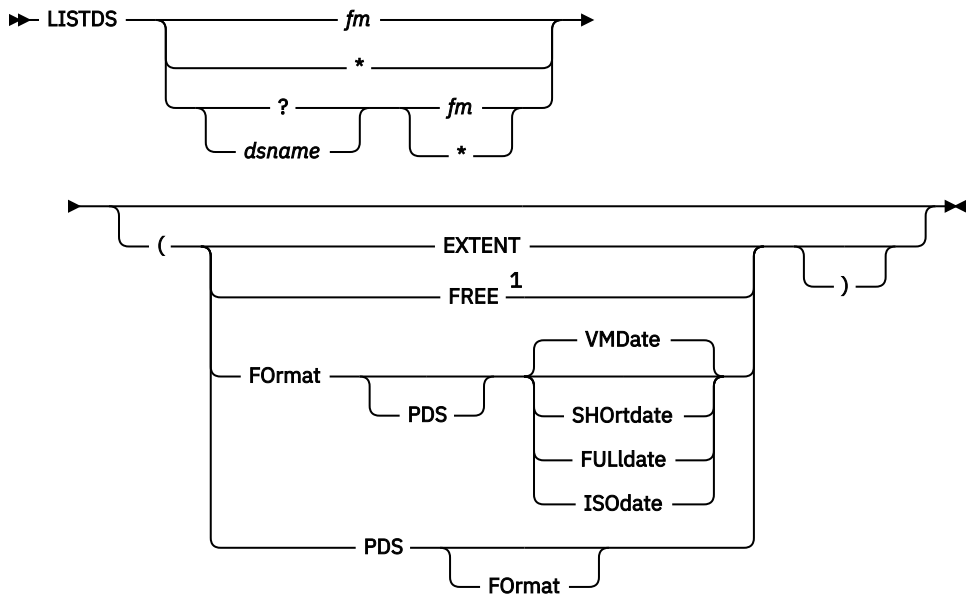
Messages and Return Codes

- DMS065E *option* option specified twice [RC=24]
- DMS066E *option1* and *option2* are conflicting options [RC=24]
- DMS069E Filemode *mode* not accessed [RC=36]
- DMS105S Error *nn* writing file to XEDIT [RC=100]
- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS688E XEDIT option only valid from XEDIT environment [RC=24]
- DMS689E File must be F-format 319 or V-format [RC=24]
- DMS1153E File pool *filepoolid* is unavailable or unknown [RC=99]
- DMS1184E Directory *dirname* not found or you are not authorized for it [RC=28]
- DMS1187E Too many subdirectory levels in *dirid* [RC=24]
- DMS1188E Filemode *mode* is not associated with a directory [RC=74]
- DMS1189E Filemode *mode* is associated with a top directory [RC=24]
- DMS1201E STACK option cannot follow FIFO or LIFO [RC=24]
- DMS1210E Directory *dirname* not found [RC=28]
- DMS1223E There is no default file pool currently defined [RC=40]
- DMS1240E You are not authorized to connect to file pool *filepoolid* [RC=76]
- DMS2524E Concurrent use of multiple file pool identifiers that resolve to file pool *filepoolid* [RC=40]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411
Errors in the Shared File System	“File Pool Server Messages” on page 1414

LISTDS



Notes:

[1] The FREE option cannot be specified with ? or *dsname*.

Authorization

General User

Purpose

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.

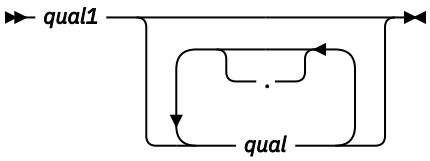
Operands

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 you want all 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 your accessed DOS or OS disks as soon as it finds the first copy of the specified data set or file.

?
indicates 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, VSE 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:



where qual1 and qual are the qualifiers of the dataset. If blanks separate the qualifiers, the dataset name used will be the concatenation of the qualifiers with periods. For more information, see Usage Note "1" on page 443.

Options

EXTENT
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, a list of all currently allocated extents on the specified disk, or on all disks, is displayed.

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.

FOrmat
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.

VMDate
displays VTOC dates using the system CP DATEFORMAT setting to determine the output format. This is the default. For more information on the user's default date format setting, see Usage Note "7" on page 443.

SHOrtdate
displays the VTOC dates in *mm/dd/yy* date format.

Where:

mm
specifies the month

dd
specifies the day of the month

yy
specifies the 2-digit year

FULldate
displays the VTOC dates in *mm/dd/yyyy* date format.

Where:

mm
specifies the month

dd
specifies the day of the month

yyyy
specifies the 4-digit year

For examples of the displays produced as a result of the FULLDATE option, see "Responses" on page 444.

ISOdate
displays the VTOC dates in *yyyy-mm-dd* date format.

Where:

yyyy
specifies the 4-digit year

mm
specifies the month

dd
specifies the day of the month

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.DATA could be listed as follows:

```
listds test input source data *
-- or --
listds test.input.source.data fm
```

   Or, you can enter the name interactively, as follows:

```
listds ? *
DMS220R  Enter data set name:
test.input.source.data
```

   Note: 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. The qualifiers of the file identifier, when directly entered from the command line, are limited to 1-8 characters between the periods or blanks. If longer length qualifiers are necessary, the interactive form must be used to enter them. The maximum length of a file identifier must not exceed 44 characters, including periods.

   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. Because CMS does not support ISAM files, LISTDS lists extent and free information on ISAM files, but ignores format 2 DSCBs.
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. For more information on how an assembler language program can supply an extended plist, see *z/VM: CMS Application Development Guide for Assembler*.
7. If a date format is not specified on the LISTDS command, the default date format is determined by the setting of the user's default date format in CP.

   The default date format for certain CP and CMS commands can be set on a system-wide basis and also for the individual user.

The system-wide default date format is set with the SYSTEM_DATEFORMAT system configuration statement. The user's default date format is set with the DATEFORMAT user directory control statement. The system-wide default and the user's default can also be set with the CP SET DATEFORMAT command. If SET DATEFORMAT is not specified for an individual user and there is no DATEFORMAT statement in the user's directory entry, the default format for that user is the system-wide default. The system-wide and user settings can be queried with the CP QUERY DATEFORMAT command.

The hierarchy of possible date format settings for the LISTDS command, from the highest priority to the lowest, is:

- LISTDS command option
- User default
- System-wide default

Responses

```
DMS220R  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
DMS220R  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.

```
DMS229I  No members found
```

This message is displayed when you use the PDS option and the data set has no members.

```
DMS233I  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 g (extent

Extent information for VTOC on G disk:
SEQ TYPE    CYL-HEAD    (RELTRK) TO   CYL-HEAD    (RELTRK)     TRACKS
000 VTOC  00099 00000       1881    00099 00018       1899         19

Extent information for PRIVAT.CORE.IMAGE.LIB ON G DISK:
SEQ TYPE    CYL-HEAD    (RELTRK) TO   CYL-HEAD    (RELTRK)     TRACKS
000 DATA  00000 00001          1    00049 00018        949        949

Extent information for SYSTEM.WORK.FILE.NO.6 on G disk:
SEQ TYPE    CYL-HEAD    (RELTRK) TO   CYL-HEAD    (RELTRK)     TRACKS
000 DATA  00050 00000        950    00051 00018        987         38

Extent information for COBOL TEST PROGRAM on G disk:
SEQ TYPE    CYL-HEAD    (RELTRK) TO   CYL-HEAD    (RELTRK)     TRACKS
000 DATA  00052 00002        990    00054 00001       1027         38

Extent information for DKSQ01A on G disk:
SEQ TYPE    CYL-HEAD    (RELTRK) TO   CYL-HEAD    (RELTRK)     TRACKS
000 DATA  00080 00001       1521    00081 00000       1539         19
```

Or for a fixed-block device:

```
Extent information for DSQ01A on G disk:
SEQ TYPE     BLOCKNO  TO    BLOCKNO          BLOCKS
000 DATA  0000000500     0000000550             51
```

Where:

SEQ
indicates the sequence number assigned this extent when the extents were defined by 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-HEAD (RELTRK) TO CYL-HEAD (RELTRK)
indicates the cylinder, head, and relative track numbers, in decimal 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, in decimal 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-HEAD    (RELTRK) TO   CYL-HEAD    (RELTRK)     TRACKS
00052 00000        988    00052 00001        989          2
00054 00002       1028    00080 00000       1520        493
00081 00001       1540    00098 00018       1880        341
```

or for a fixed-block device:

```
listds g (free

Freespace extents for G disk:
FB/E BLOCKNO  TO  FB/E BLOCKNO      BLOCKS
0000000501        0000001330         830
```

```
0000010310        0000029610       19301
0000068990        0000069990        1001
```

Where:

CYL-HEAD (RELTRK) TO CYL-HEAD (RELTRK)
indicates the cylinder, head and relative track numbers, in decimal of the starting and ending track in the free extent.

TRACKS
indicates the total number of free tracks in the extent.

FB/E BLOCKNO TO FB/E BLOCKNO
indicates the relative block numbers, in decimal 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, PDS and FULLDATE options, you receive information similar to this:

```
listds d (fo pds ful)

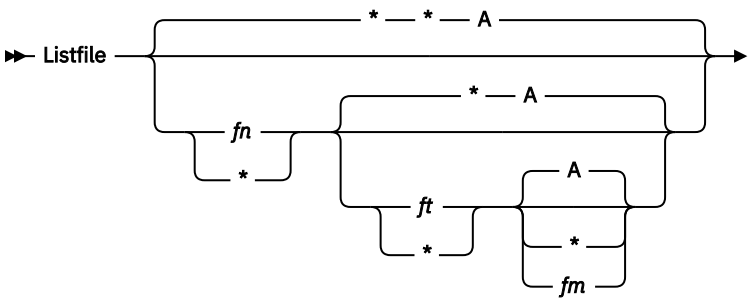
RECFM LRECL BLKSI DSORG   DATE      LABEL   FM  DATA SET NAME
FB    80    800   PO    11/14/1997  OSSYS1  D   SYS1.MACLIB
MEMBER NAMES:
ABEND    ATTACH   BLDL     BSP       CLOSE  DCB  DETACH   DEVTYPE
FIND     PUT      READ     WRITE     XDAP

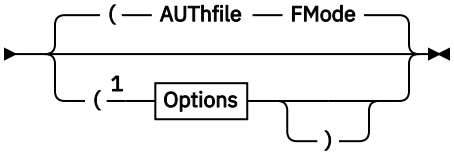
RECFM LRECL BLKSI DSORG   DATE       LABEL   FM  DATA SET NAME
F     80    80    PS    02/25/1989   OSSYS1  D   SAMPLE
```

Messages and Return Codes

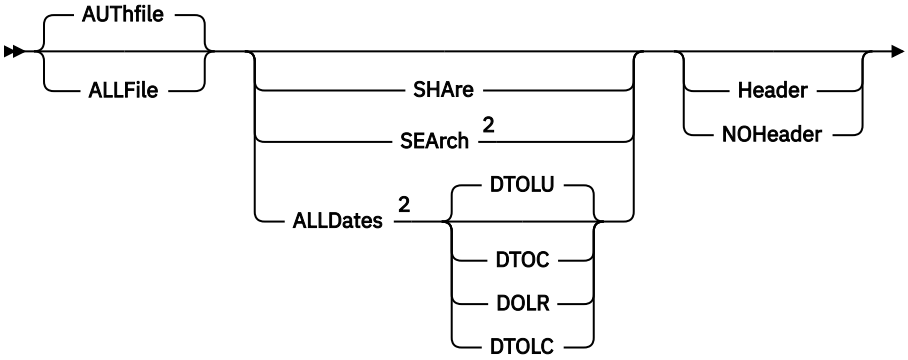
- DMS002E Dataset not found [RC=28]
- DMS003E Invalid option: *option* [RC=24]
- DMS048E Invalid filemode *mode* [RC=24]
- DMS069E Filemode *mode* not accessed [RC=36]
- DMS221E Invalid data set name [RC=24]
- DMS222E I/O error reading *data set name* from {*fm*|OS|DOS} disk [RC=28]
- DMS223E No filemode specified [RC=24]
- DMS226E No dataset name allowed with FREE option [RC=24]
- DMS227W Invalid extent found for *data set name* on *fm* disk [RC=4]
- DMS231E I/O error reading VTOC from {*fm*|OS | DOS} disk [RC=28]
- DMS333E *nnnnn*K partition too large for this virtual machine [RC=24]

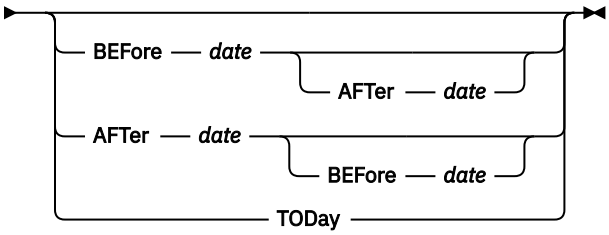
LISTFILE





Options



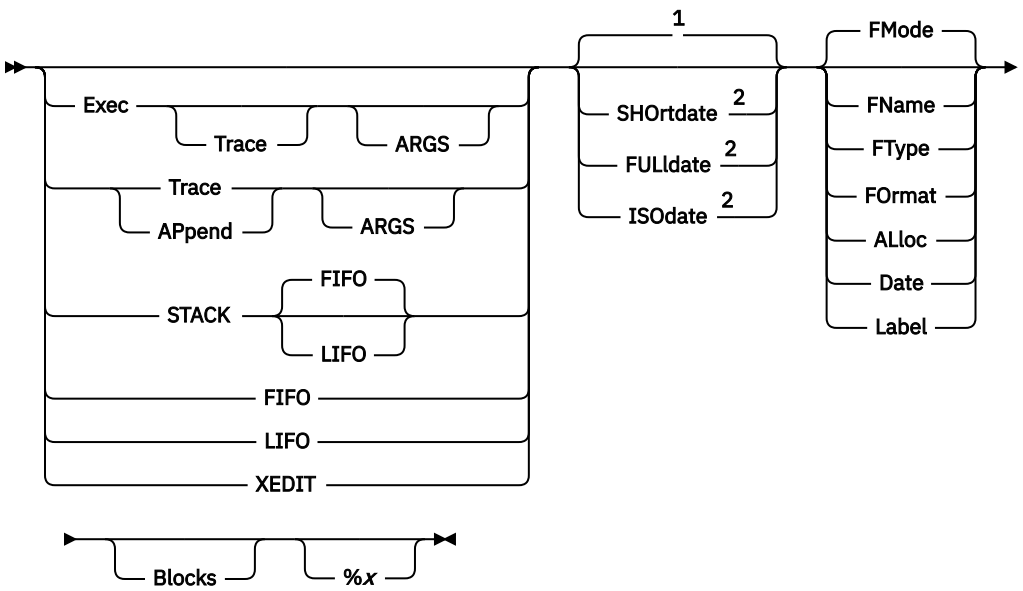


Notes:

[1] You can enter Options in any order between the parentheses.

[2] If this option is chosen, you cannot use the BEFORE, AFTER, or TODAY options.

Options



Notes:

[1] If a date format is not specified, the user's default date format is used when the DATE or ALLDATES option is specified.

[2] If a date format of SHORTDATE, FULLDATE, or ISODATE is specified and the DATE or ALLDATES option is not specified, LISTFILE operates as if the DATE option was specified.

Authorization

General User

Purpose

Use the LISTFILE command to obtain specified information about:

- CMS files residing on accessed minidisks
- Files and subdirectories in Shared File System (SFS) directories that are accessed

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. This is the default.

In addition, certain special characters (* and %) can be used as part of the file name to request the list contain a specific subset of files. For more information, see "Using Pattern Matching to Specify Sets of Files" on page 14.

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. This is the default.

In addition, certain special characters (* and %) can be used as part of the file type to request the list contain a specific subset of files. For more information, see "Using Pattern Matching to Specify Sets of Files" on page 14.

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. This is the default. If an asterisk is coded, all accessed disks and directories are searched.

Options

Shared File System (SFS) Options

You can use the ALLFILE, AUTHFILE, SHARE, SEARCH, and ALLDATES 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 and ALLDATES options will also specify the information you want about files on disks.

ALLFile
lists the specified files in a directory whether you have been granted read or write authority on them. Subdirectories, erased or revoked aliases, and external objects are also listed.

AUThfile
lists only the specified files in a directory you have been granted read or write authority to. Subdirectories, erased or revoked aliases, and external objects are **not** listed. The default is AUTHFILE.

SHAre
lists the following information about the specified files (or subdirectories, erased or revoked aliases, and external objects only if the ALLFILE option is also specified):

- File or directory identifier
- Owner
- Type (minidisk, directory, base, alias, erased or revoked alias, or external object)
- What authority you have to the object (R/W)

The SHARE option cannot be specified with the following options:

- FORMAT
- ALLOC
- DATE
- LABEL
- SEARCH
- ALLDATES

If you use the SHARE option on a DIRCONTROL directory you have accessed R/O, the authorizations for all the files in that directory will be READ. This is true even if you have directory control write (DIRWRITE) authority to the directory. When you have the directory accessed R/O, you are not able to write to any files. So, read authorization is displayed for every file. To view your true authority, use QUERY AUTHORITY or AUTHLIST commands.

For performance reasons, a question mark (?) is sometimes returned in the "R" and "W" columns. The question mark is returned only for subdirectories when:

- The parent directory has the DIRCONTROL attribute, and
- An External Security Manager (ESM) is not being used, and
- You are not the owner of the subdirectory.

To view your authorities, use the AUTHLIST or QUERY AUTHORITY commands.

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 locked EXCLUSIVE by another user are not searched. If you do not specify *fm*, the search begins at the 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
- ALLDATES
- BEFORE
- AFTER
- TODAY

ALLDates
Lists these dates and times:

- Date of creation
- Time of creation
- Date of last reference
- Date of last update
- Time of last update
- Date of last change (with DTOLC option)
- Time of last change (with DTOLC option)

The ALLDATES option cannot be specified with the following options:

- FORMAT
- ALLOC
- DATE
- LABEL
- SEARCH
- SHARE
- BEFORE
- AFTER
- TODAY

The ALLDATES option displays the date and time of creation, the date of last reference, the date and time of last update, and (with the DTOLC option) the date and time of last change for directory objects. For minidisk files, subdirectories, erased aliases, and revoked aliases, a dash appears in the date of creation (`Create-Dt`) column, the time of creation (`Create-Tm`) column, and the date of last

reference (`Lref-Dt`) column. For minidisk files, a dash appears in the date of last change (`Lchng-Dt`) column and the time of last change (`Lchng-Tm`) column. For external objects, a dash appears in the date of last reference (`Lref-Dt`) column. For more information on each column displayed by the LISTFILE command, see Responses.

The DTOC, DOLR, DTOLU, and DTOLC options determine which fields are in the records produced by LISTFILE (ALLDATES. The options are assigned priorities, and each option includes the options with higher priority numbers.

DTOLU
displays the date and time the file, alias, or external object was last updated in one of the following formats:

```
mm/dd/yy  hh:mm:ss
mm/dd/yyyy hh:mm:ss
yyyy-mm-dd hh:mm:ss
```

Priority is 2. The default is DTOLU.

DTOC
Displays the date and time of the object's creation in one of the following formats:

```
mm/dd/yy  hh:mm:ss
mm/dd/yyyy hh:mm:ss
yyyy-mm-dd hh:mm:ss
```

Priority is 4.

DOLR
displays the date of last reference to the object in one of the following formats:

```
mm/dd/yy
mm/dd/yyyy
yyyy-mm-dd
```

Priority is 3.

DTOLC
displays the date and time of last change for an object in one of the following formats:

```
mm/dd/yy  hh:mm:ss
mm/dd/yyyy hh:mm:ss
yyyy-mm-dd hh:mm:ss
```

Priority is 1. For more information, see Usage Note "14" on page 458.

Output Format Options

Header
includes column headings in the listing. The default is HEADER if any of the supplemental information options (FORMAT, ALLOCATE, SHARE, SEARCH, ALLDATES, DATE, or LABEL) are specified.

NOHeader
does not include column headings in the list. NOHEADER is the default if only file name, file type, or file mode information is requested, or if you specify STACK, FIFO, or LIFO.

BEFore *date*
lists only those files last written to *before* the date specified. The date can be specified as *mm/dd/yy*, *mm/dd/yyyy*, or *yyyy-mm-dd*.

Where:

mm
specifies the month

dd
specifies the day of the month

yy
specifies the 2-digit year

yyyy
specifies the 4-digit year

The variables *mm, dd,* and *yy* are one or two digit numbers.

The BEFORE option cannot be specified with the following options:

- SEARCH
- ALLDATES

AFTer *date*
lists only those files last written to *after* the date specified. The date can be specified as *mm/dd/yy*, *mm/dd/yyyy*, or *yyyy-mm-dd*

Where:

mm
specifies the month

dd
specifies the day of the month

yy
specifies the 2-digit year

yyyy
specifies the 4-digit year

The variables *mm, dd,* and *yy* are one or two digit numbers.

The AFTER option cannot be specified with the following options:

- SEARCH
- ALLDATES

Note: When both the BEFORE and AFTER options are used:

- If the AFTER date precedes the BEFORE date, the output is the set of files last written to between the two dates.
- If the BEFORE date precedes the AFTER date, the output is the set of files last written to either before the BEFORE date or after the AFTER date; the files last written to between the two dates are excluded.
- If the same date is specified for BEFORE and AFTER, the output is the set of files last written to on any date *except* the specified date.

TODay
Lists only those files last written to on the present day.

The TODAY option cannot be specified with the following options:

- SEARCH
- ALLDATES

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. However, if no matches are found and a CMS EXEC already exists, it is erased.

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, refer the z/VM HELP Facility for EXEC 2 by entering:

```
help exec2 &trace
```

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 with EXEC, TRACE, or APPEND.

The ARGS option cannot be specified with the following options:

- FORMAT
- ALLOC
- DATE
- LABEL
- SEARCH
- SHARE
- ALLDATES

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
specifies 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 is STACK FIFO.

FIFO
specifies 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 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 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 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 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 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.

When the ALLDATES 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 174 or 400. When the logical record length is 174, each line is 174 characters long and contains the information obtained with the ALLDATES DTOLC option (file name, file type, file mode, creation date, creation time, last reference date, last update date, last update time, last change date, and last change time), plus the file name, file type, file mode, records, blocks, creation date, creation time, and last reference date. When the logical record length is 400, each line is 400 characters long and contains the information returned by the ALLDATES DTOLC option, plus the file name, file type, file mode, records, blocks, creation date, creation time, last reference date, last update date, last update time, last change date, and last change time.

SHOrtdate
displays the dates in *mm/dd/yy* format.

Where:

mm
specifies the month

dd
specifies the day of the month

yy
specifies the 2-digit year

This option is ignored if specified with the SEARCH or SHARE option.

FULldate
displays the dates in *mm/dd/yyyy* format.

Where:

mm
specifies the month

dd
specifies the day of the month

yyyy
specifies the 4-digit year

This option is ignored if specified with the SEARCH or SHARE option.

ISOdate
displays the dates in *yyyy-mm-dd* format.

Where:

yyyy
specifies the 4-digit year

mm
specifies the month

dd
specifies the day of the month

This option is ignored if specified with the SEARCH or SHARE option.

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. This is the default. Option priority is 5.

Supplemental Information Options

FOrmat
includes the record format and logical record length of each file in the list. The option priority is 4.

ALloc
includes the amount of disk space 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. The option priority is 3.

Date
includes the date the file was last written in the list in one of these formats:

```
mm/dd/yy  hh:mm:ss
mm/dd/yyyy hh:mm:ss
yyyy-mm-dd hh:mm:ss
```

Option priority is 2.

Label
includes the label of the disk on which the file resides in the list. The 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.

Note: The BLOCKS value may not reflect the total amount of space needed to store the file. For SFS files, you can get this information by using the QUERY BLOCKS command.

%x
specifies the change 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 "Using Pattern Matching to Specify Sets of Files" on page 14.

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. 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,

   ```
   PHILLIPS NOTEBOOK
   PHILLIPSNOTES
   ```

   Where:

   PHILLIPS NOTEBOOK is a file and PHILLIPSNOTES is a subdirectory, enter:

   ```
   listfile phil* n* a (allfile
   ```

   would find both PHILLIPS NOTEBOOK and PHILLIPSNOTES because PHILLIPSNOTES contains more than eight characters and is matched as if it had a file name of PHILLIPS and a file type of NOTES, enter:

   ```
   listfile phillipsnotes * a (allfile
   ```

would also find both PHILLIPS NOTEBOOK and PHILLIPSNOTES because the LISTFILE command only uses the first eight characters as the file name. Therefore, any file with the name of PHILLIPS, or any subdirectory with PHILLIPS as the first eight characters in its name would be listed.

However, if an explicit file name or file type is used, only the file matching it is returned, and no subdirectories. This is to ensure only one match is found. For example, enter:

```
listfile phillips notes (allfile
```

will result in no files being found.

3. 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.
4. If you request any additional information with the supplemental information options, that information is displayed along with the header.
5. 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 you enter:

   ```
   LISTFILE * * A (EXEC ARGS
   ```

   a CMS EXEC file is created, with each record formatted.

   Example 1

   If you enter:

   ```
   cms tape dump ( wtm
   ```

   this 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.

   Example 2

   If you enter:

   ```
   cms copyfile % = save a
   ```

   this causes the copyfile command to be executed against each file in the CMS EXEC, with COPYFILE assigned to &1, = to &3, save to &4, and a to &5. (&2 is skipped because % is the placeholding character.) In this case, this results in the following command being issued for each file in the exec:

   ```
   copyfile fn ft = save a
   ```

   If you use any of the supplemental information options, that information is included in the EXEC file.
6. 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 REXX exec with ADDRESS COMMAND in effect.

7. 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
   ```

8. 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.
9. The default date format for certain CP and CMS commands can be set on a system-wide basis and also for the individual user. The system-wide default date format is set with the SYSTEM_DATEFORMAT system configuration statement. The user's default date format is set with the DATEFORMAT user directory control statement. The system-wide default and the user's default can also be set with the CP SET DATEFORMAT command. If SET DATEFORMAT is not specified for an individual user and there is no DATEFORMAT statement in the user's directory entry, the default format for that user is the system-wide default. The system-wide and user settings can be queried with the CP QUERY DATEFORMAT command. The hierarchy of possible date format settings for the LISTFILE command, from highest priority to lowest, is:
   - LISTFILE command option
   - User default
   - System-wide default
10. The dates and times displayed for an alias are the same as the dates and times displayed for its base file. However, the date of last reference of an alias is not the same as the date of last reference of its base file. The date of last reference of an alias does not get updated when either the alias or its base file is referenced. At the time of creation of an alias, the date of last reference value for that alias takes on the date of last reference value of the base file. This alias date of last reference value does not change.
11. Dates and times displayed are local dates and times, except for the date of last reference, which is in Coordinated Universal Time (UTC).

    When the creation date or time is not a valid date or time value, the conversion to local time does not take place and they are left in UTC.

    The month portion, *mm*, of the dates displayed in both the SHORTDATE format and the FULLDATE format contains a single digit for months 1-9 (January to September). The month portion of the dates displayed in the ISODATE format contains two digits, and for months 1-9 (January to September) the month displayed contains a leading zero.
12. For the BEFORE and AFTER options, if you are running a release prior to VM/ESA version 2 release 2, and you specify a date of *mm/dd/yy*, the sliding window technique calculates a 4-digit year (*yyyy*) from the 2-digit year, (*yy*) that is input.

    The calculation assumes the 2-digit year is within the 100 year window as calculated by:

    ```
    (current_year - 50) = low end of window
    (current_year + 49) = high end of window
    ```

    For example, if a 2-digit year of 05 is supplied, and the current year (current_year;) is 1997, the window range is between the years 1947 and 2046. In this case, the calculation changes the 2-digit year of 05 to the 4-digit year 2005.
13. To display files using the BEFORE/AFTER/TODAY options, enter:

```
listfile * * a (after mm/dd/yy before mm/dd/yy
listfile * * b (before mm/dd/yy
listfile * * c (today
```

or

```
listfile * * a (after mm/dd/yyyy before mm/dd/yyyy
listfile * * b (before mm/dd/yyyy
listfile * * c (today
```

or

```
listfile * * a (after yyyy-mm-dd before yyyy-mm-dd
listfile * * b (before yyyy-mm-dd
listfile * * c (today
```

For example, to display all files between Jan 1, 1991, and Feb 1, 1991, enter:

```
listfile * * a (after 12/31/90 before 2/02/91
```

or

```
listfile * * a (after 12/31/1990 before 2/02/1991
```

or

```
listfile * * a (after 1990-12-31 before 1991-02-02
```

14. The date and time of last change are maintained by SFS to support DFSMS/VM. They cannot be changed by users, administrators, or applications. The date and time of last change reflect changes to an object's existence, data, attributes, authorizations, name, and location; changes to space limits are also reflected for file spaces.

    The date and time of last change are not updated:

    for a file
    when it is migrated or restored

    for a directory
    when an object is added or deleted

    for an alias
    when the base file is updated.

For more information about updates to the date of last reference, see *z/VM: CMS Application Development Guide*.

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, the information displayed is:

```
FILENAME FILETYPE  FM  FORMAT LRECL RECS   BLOCKS      DATE     TIME   LABEL
   fn       ft     fm  format lrecl norecs noblks  mm/dd/yyyy hh:mm:ss volid
   .        .      .   .      .     .      .       .          .        .
   .        .      .   .      .     .      .       .          .        .
   .        .      .   .      .     .      .       .          .        .
```

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, and DIR is a directory. A dash indicates an erased or revoked alias, or external object.

lrecl
is the logical record length of the largest record in the file. For directories, revoked or erased aliases, and external objects, a dash is displayed.

norecs
is the number of logical records in the file. For directories, revoked or erased aliases, and external objects, a dash is displayed. For empty files, a zero is displayed.

noblks
is the number of CMS data blocks the file occupies. (In some cases, this number represents logical rather than physical data blocks, and can exceed the actual number of blocks allocated to the file.) Also, this number may not represent the total number of blocks needed to store the file. For SFS files, you can get this information by using the QUERY BLOCKS command. For directories, revoked or erased aliases, and external objects, a dash is displayed. For empty files, a zero is displayed.

Note: For sparse files with a record format of:

- Fixed, the BLOCKS value does not include the empty blocks
- Variable, the BLOCKS value does include the empty blocks

A sparse file is a file with one of more blocks filled with binary zeroes. CMS does not write such blocks physically on the disk.

mm/dd/yyyy
is the date (month/day/year) the file was last updated.

Note: The FULLDATE was in effect for this example. 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) 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.

volid
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
  .         .     .  .        .       . .
  .         .     .  .        .       . .
  .         .     .  .        .       . .
```

owner
is the user ID of base file owner. If the type is base file or alias, it is the user ID of the base file owner. For files on disks, this will be the disk owner's user ID. If the type is directory, it is the user ID of the owner of the base file which was erased. If the type is revoked alias, it is the user ID of the owner of the base file which has had all authorizations revoked for the owner of the directory containing the alias. If the type is external object, it is the user ID of the owner of the directory containing the external object.

type
is one of the following:

- MDISK for a file on a minidisk
- BASE for a base file in a directory

- DIR for a FILECONTROL directory
- DIRC for a DIRCONTROL directory
- ALIAS for an alias in a directory
- ERASED for an erased alias
- REVOKED for a revoked alias
- ALIAS* for an alias of a file that is in DFSMS/VM migrated status
- BASE* for a base file that is in DFSMS/VM migrated status
- EXTRNL for an external object

r
is read authority.

X
indicates you have read authority.

–
A dash indicates you do not have read authority.

P
means the authority is managed by an External Security Manager (ESM).

?
means SFS cannot readily determine the authorization.

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 you have write authority.

–
A dash indicates you do not have write authority.

P
means the authority is managed by an External Security Manager (ESM).

?
means SFS cannot readily determine the authorization.

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
  .         .     .  .
  .         .     .  .
  .         .     .  .
```

directory
is the complete name of the directory that contains the file.

One entry is displayed for each file listed. If the specified files are on a disk, you receive the message:

```
DMSWFL1182E  The SEARCH option may not be used with a minidisk
Update-TM LCHNG
13:14:53    06/18
DT  LCHNG-TM
1996  19:30:03
```

If the ALLDATES and FULLDATE options are specified, the information displayed is:

```
Filename Filetype Fm Create-Dt   Create-Tm   Lref-Dt  Update-Dt   Update-Tm
fn       ft       fm mm/dd/yyyy  hh:mm:ss  mm/dd/yyyy mm/dd/yyyy  hh:mm:ss
```

```
.        .        .  .          .         .          .          .
.        .        .  .          .         .          .          .
.        .        .  .          .         .          .          .
```

fn
is the name of the file, external object or directory.

ft
is the file type of the file or external object. For a directory this column is blank.

fm
is the file mode of the file, external object, or directory.

Create-Dt mm/dd/yyyy
is the date (month/day/year) the file was created. A dash appears in this column for erased aliases, revoked aliases, subdirectories, and minidisk files. This column contains 00/00/0000 if the file pool server does not support the creation date.

Create-Tm hh:mm:ss
is the time (hours:minutes:seconds) the file was created. A dash appears in this column for erased aliases, revoked aliases, subdirectories, and minidisk files. This column contains 00:00:00 if the file pool server does not support the creation time.

Lref-Dt mm/dd/yyyy
is the date (month/day/year) the file was last referenced. A dash appears in this column for erased aliases, revoked aliases, external objects, subdirectories, and minidisk files. This column contains 00/00/0000 if the file pool server does not support the date of last reference.

Update-Dt mm/dd/yyyy
is the date (month/day/year) the file was last updated. For a subdirectory, this is the date the subdirectory was created. A dash appears in this column for erased or revoked aliases.

Update-Tm hh:mm:ss
is the time (hours:minutes:seconds) the file was last updated. For a subdirectory, this is the time the subdirectory was created. A dash appears in this column for erased or revoked aliases.

When the ALLDATES DTOLC and FULLDATE options are used, the information displayed is:

```
Filename Filetype Fm Create-Dt   Create-Tm   Lref-Dt  Update-Dt  Update-Tm Lchng-Dt     Lchng-Tm
fn       ft       fm mm/dd/yyyy  hh:mm:ss  mm/dd/yyyy mm/dd/yyyy hh:mm:ss  mm/dd/yyyy   hh:mm:ss
.        .        .  .           .         .          .          .         .            .
.        .        .  .           .         .          .          .         .            .
.        .        .  .           .         .          .          .         .            .
```

Note: When this information is displayed on your screen, your screen size settings may cause the information to wrap.

The first eight columns are the same as described in the ALLDATES example,

Lchng-Dt mm/dd/yyyy
is the date of last change (month/day/year) for the file, alias, external object, or directory. A dash appears in this column for erased or revoked aliases and minidisk files. This column contains 00/00/0000 if the file pool server does not support the date of last change.

Lchng-Tm hh:mm:ss
is the time of last change (hours:minutes:seconds) for the file, alias, external object, or directory. A dash appears in this column for erased or revoked aliases and minidisk files. This column contains 00:00:00 if the file pool server does not support the time of last change.

For more information on the LISTFILE command, see *z/VM: CMS User's Guide*.

Messages and Return Codes

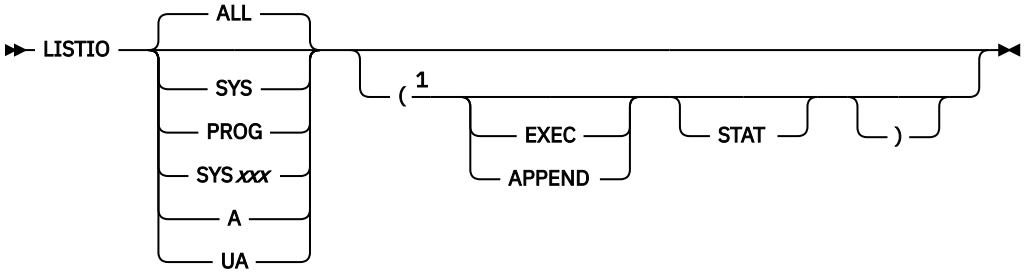
- DMS002E File *fn ft fm* not found [RC=28]
- DMS003E Invalid option: *option* [RC=24]
- DMS037E Filemode *mode* is accessed as read-only [RC=36]

- DMS048E Invalid filemode *mode* [RC=24]
- DMS066E *option1* and *option2* are conflicting options [RC=24]
- DMS069E Filemode *mode* not accessed [RC=36]
- DMS070E Invalid parameter *parameter* [RC=24]
- DMS105S Error *nn* writing file *fn ft fm* to XEDIT [RC=100]
- DMS109E Virtual storage capacity exceeded [RC=104]
- DMS688E XEDIT option only valid from XEDIT environment [RC=24]
- DMS689E File must be F-format *lrecl* or V-format [RC=24]
- DMS765E Incorrect date specified [RC=24]
- DMS765E No files matched the specified date range [RC=28]
- DMS1182E The SEARCH option may not be used with a minidisk [RC=74]
- DMS1183E '*' may not be specified for the filemode with the SEARCH option [RC=24]
- DMS1184E Directory *dirname* not found or you are not authorized for it [RC=28]
- DMS2059E DTOC, DOLR, DTOLU & DTOLC can not be specified without the ALLDATES option.

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	"Command Syntax Error Messages" on page 1411
Errors in the Shared File System	"File Pool Server Messages" on page 1414

LISTIO



Notes:

[1] You can enter Options in any order between the parentheses.

Authorization

General User

Purpose

Use the LISTIO command in CMS/DOS to display a list of current assignments for system and programmer logical units in your virtual machine.

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.

SYS*xxx*
requests a display of the physical device assigned to the particular logical unit specified.

Where:

xxx
specifies the 3-digit number of a logical unit.

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. The default is ALL.

Options

The EXEC and APPEND options are mutually exclusive; if both are entered on the command line, the last one entered is in effect.

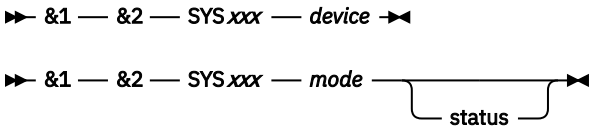
EXEC
erases the existing $LISTIO EXEC file, if one exists, and creates a new one.

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 more information on logical units and valid device types, see "ASSGN" on page 53.
2. The $LISTIO EXEC contains one record for each logical unit listed. The format is:



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.

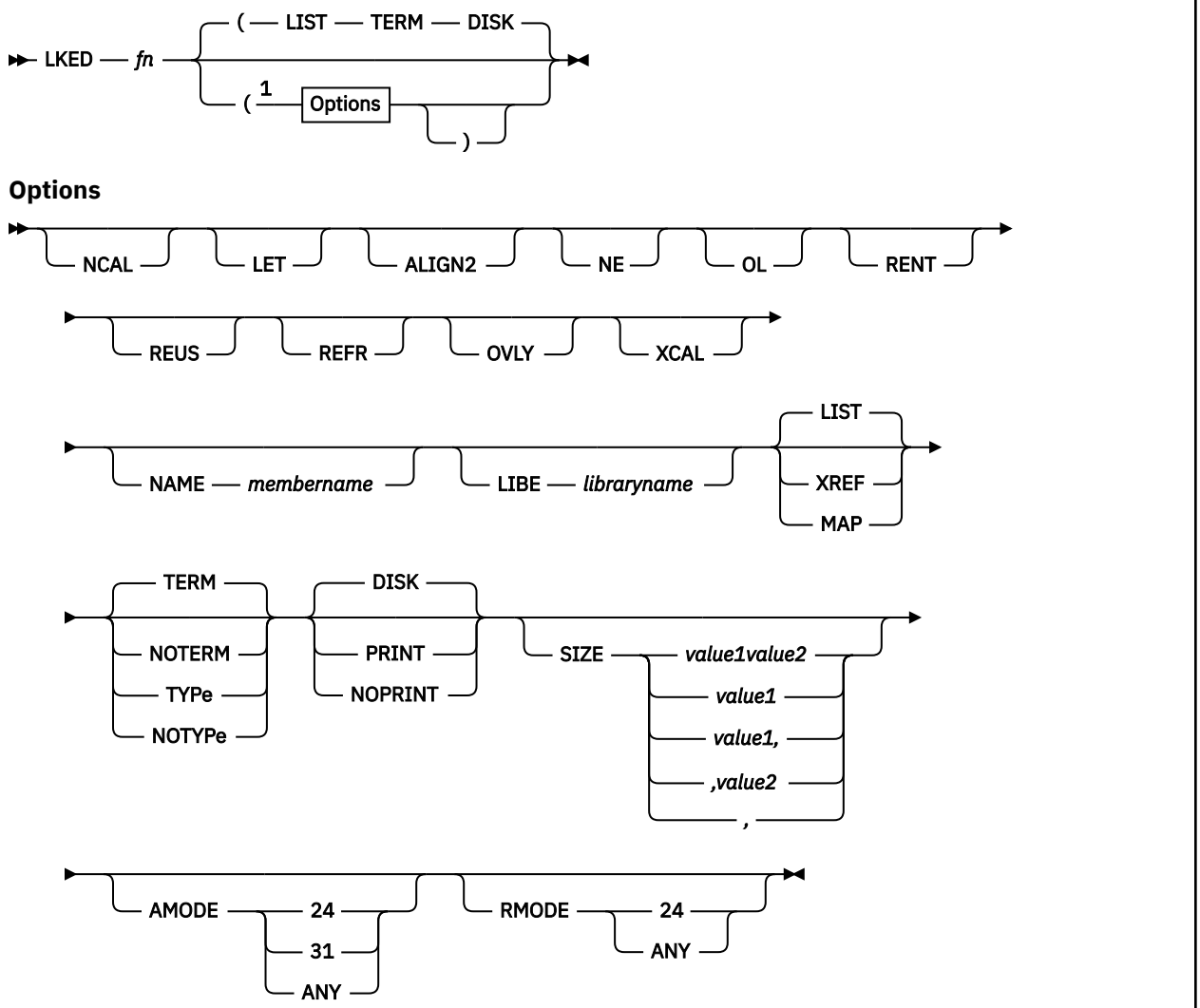
Messages and Return Codes

- DMS003E Invalid option: *option* [RC=24]
- DMS006E No read/write A filemode accessed [RC=36]
- DMS070E Invalid parameter *parameter* [RC=24]
- DMS099E CMS/DOS environment not active [RC=40]
- DMS105S Error *nn* writing file *fn ft fm* on disk or directory [RC=100]
- DMS257T Internal system error at address *address* (offset *offset*)
- DMS303E No SYSXXX satisfies request [RC=28]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in the Shared File System	"File Pool Server Messages" on page 1414
Errors in using a file	"File Error Messages" on page 1418

LKED



Notes:

[1] You can enter Options in any order between the parentheses.

Authorization

General User

Purpose

Use the LKED command to create a CMS LOADLIB or LOADLIB member.

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 the procedures. 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, TYPE, NOTYPE, PRINT, DISK, NOPRINT, NAME, LIBE.

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 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 re-enterable.

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 by means of the linkage editor NAME control statement. The default member name is the same as the *fn* specified.

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 by means of 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. This is the default.

TERM
displays any linkage editor diagnostic messages at the user terminal. This is the default.

NOTERM
suppresses the displaying of diagnostic messages.

TYPe
displays any linkage editor diagnostic messages at the user terminal. (This option has the same function as the TERM option.)

NOTYPe
suppresses the displaying of diagnostic messages. (This option has the same function as the NOTERM option.)

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. This is the default.

NOPRINT
produces no output file.

SIZE *value1 value2*
indicates the amount of virtual storage to be used by the linkage editor (*value1*) and specifies the portion of that storage to be reserved for the load module buffer (*value2*). The SIZE values must lie within the following limits:

value1
64K to 9999K (or 65536 to 999999)

value2
6K to 100K (or 6144 to 102400)

If a SIZE value is omitted, the default is 400K for *value1* or 100K for *value2*. If a specified value is not valid, the OS Linkage Editor defaults to to the maximum value, which is 9999K for *value1* or 100K for *value2*. For *value1*, values greater than 999999 must be entered in the form *nnnn*K (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. 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.

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 the location in virtual machine storage where the loaded module is to reside.

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, overriding the RMODE determined by the linkage editor.

ANY
The load may reside above or below the 16MB line, overriding the RMODE determined by the linkage editor.

Usage Notes

1. Only a subset of the possible linkage editor control statements are meaningful in CMS. Because the CMS interface program cannot examine the input data for the LKED command, all 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. All command options and control statements not meaningful to CMS are simply passed to the linkage editor without validation.
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 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) 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 MVS 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.

The default FILEDEF commands issued by the LKED command for the *ddnames* 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 the SYSPRINT FILEDEFs are specified without the NOCHANGE option. This way the filedefs are not overridden. The *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 block size of the output loadlib defaults to 260. The default can be overridden if the user issues a FILEDEF statement prior to the LKED command and specifies DDNAME=SYSLMOD. If the:
   - FILEDEF command is issued without a block size, the system default is 32760.
   - FILEDEF command specifies a block size <= 32760, the block size is set to the specified value.
   - Disk already has a file with the same FN and FT as the DISK output file specified in the FILEDEF statement, the block size of that file will remain the same.
8. For information on listing, copying, and compressing a CMS LOADLIB, see "LOADLIB" on page 487.
9. You can use the FILEDEF command with the CONCAT option and the GLOBAL command to concatenate TXTLIBs as input to the LKED command. For example:

   ```
   GLOBAL TXTLIB TXLIB1 TXLIB2
   FILEDEF SYSLIB DISK TXLIB1 TXTLIB A (CONCAT
   LKED fn
   ```

   The *ddname* on the FILEDEF command must be SYSLIB. The file name on the FILEDEF command can be any valid TXTLIB file name. (This file name does not have to correspond to any file names listed on the GLOBAL command.) The file type must be TXTLIB, and the file must be a fixed-formatted file.

   When concatenating TXTLIBs, all the TXTLIBs listed on the GLOBAL command are searched and the TXTLIB specified on the FILEDEF command is ignored. The GLOBAL command determines the order in which the libraries are searched. For more information on concatenating TXTLIBs as input to the LKED command, see *z/VM: CMS Application Development Guide for Assembler*.
10. When TEXT files to be linked contain direct calls to CSL routines the names of the CSL libraries, from which the CSL routines come, must be included in the GLOBAL TXTLIB command preceding the LKED command.

    You can determine whether a CSL routine is a direct call routine by performing a CSLLIST, specifying the library name or performing CSLMAP *rtname* (ALL.

Messages and Return Codes

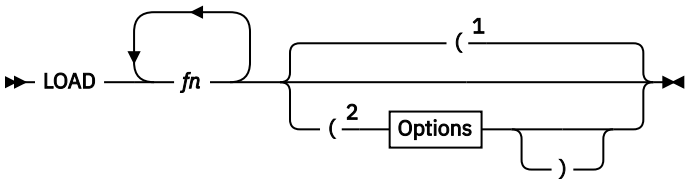
- DMS001E No filename specified [RC=24]

- DMS002E File *fn [ft [fm]]* not found [RC=28]
- DMS004W Warning messages issued [RC=4]
- DMS005E No *option* specified [RC=24]
- DMS006E No read/write filemode accessed [RC=36]
- DMS007E File *fn ft fm* [is] not fixed, 80-character records [RC=32]
- DMS008W Error messages issued [RC=8]
- DMS012W Severe error messages issued [RC=12]
- DMS016W Terminal error messages issued [RC=16]
- DMS070E Invalid parameter *parameter* [RC=24]
- DMS1229E *fileid* is empty [RC=88]

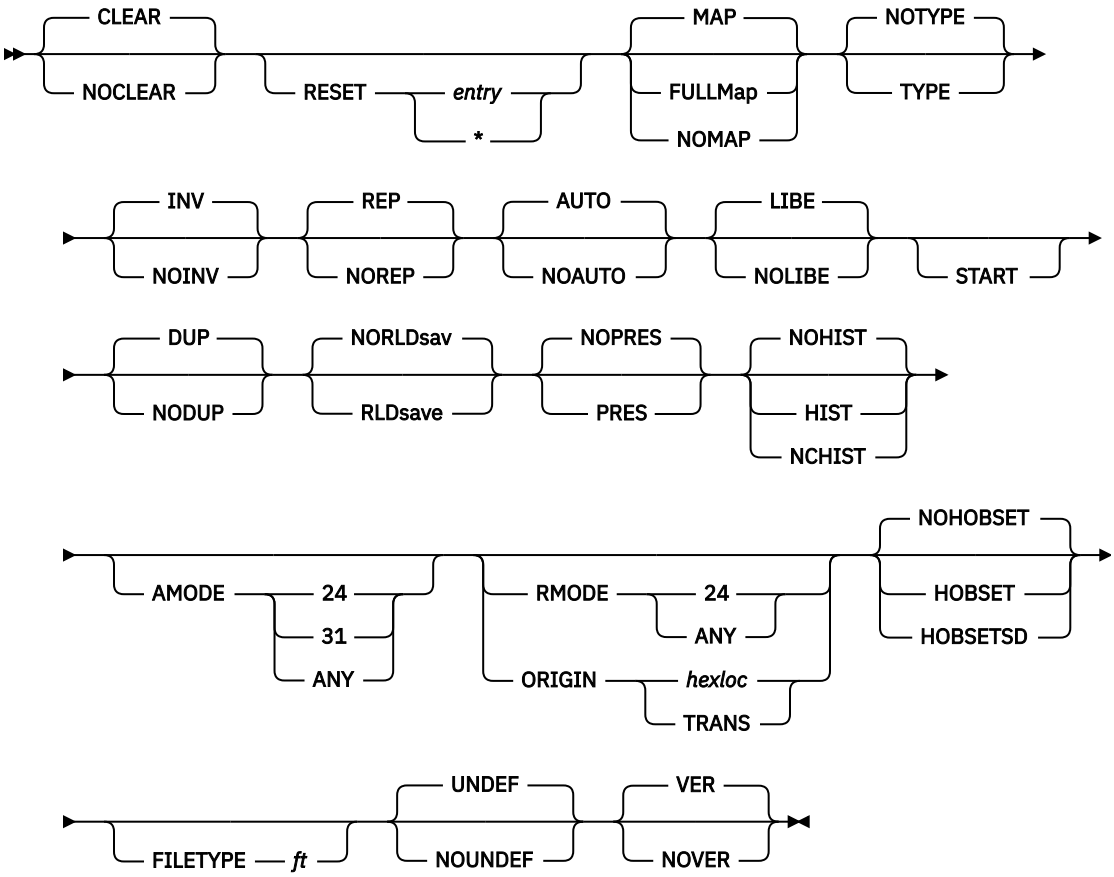
Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in erasing a file	“Messages and Return Codes” on page 217

LOAD



Options



Notes:

[1] The defaults you receive appear above the line in the Options fragment.
[2] You can enter Options in any order between the parentheses.

Authorization

General User

Purpose

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.

- Use them as input to GENMOD to create MODULE files

- Execute them using the START command

Note: The SET LOADAREA command affects the outcome of the LOAD and INCLUDE commands.

Operands

fn
specifies the names of the files to be loaded into storage. The default file type is TEXT unless the FILETYPE option specifies the file type. Files must 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
sets storage to binary zeros before loading. For compatibility only. The default is CLEAR.

NOCLEAR
does not clear the load area before loading. NOCLEAR is only valid when loading into the transient area.

RESET *entry*
RESET *
sets the starting location for the programs currently loaded. If *entry* is specified, the starting address is reset to the specified location. 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. For more information, see Usage Note "3" on page 477. 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. The default is MAP.

FULLMap
writes a load map on your minidisk or directory accessed as A named *fn* LOADMAP A5, where *fn* is the first file name specified on LOAD. The load map file contains full information and is of variable record length.

NOMAP
does not create a load map file.

TYPE
displays the load map at your terminal.

NOTYPE
does not display the load map at the terminal. The default is NOTYPE.

INV
includes not valid card images in the load map. The default is INV.

NOINV
does not include card images that are not valid in the load map.

REP
includes Replace (REP) statements in the load map. The default is REP.

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. The default is AUTO.

NOAUTO
suppresses automatic searching for text files.

LIBE
searches the text libraries defined by a previously issued GLOBAL command for unresolved references. The default is LIBE.

NOLIBE
does not search the text libraries for unresolved references.

START
immediately executes the program being loaded upon successful completion of the load. LOAD does not generally begin execution of the loaded files. Execution begins at the default entry point. For more information, see Usage Note "3" on page 477.

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. The default is DUP. For more information, see Usage Note "2" on page 477.

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. The default is NORLDsav.

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 the currently loaded set of programs occupies. These programs can no longer be referenced by another program. The default is NOPRES.

PRES
instructs CMS not to release the storage 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.

NOHIST
does not save history information from text files. The default is NOHIST.

HIST
saves the history information 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.

NCHIST
saves only uncommented history records for inclusion in the module generated by a GENMOD command. Commented history records (that is, those with an asterisk (*) in column one) will be ignored.

AMODE
specifies the addressing mode in which the program will be entered. The AMODE defined by this option propagates to the GENMOD process.

This option overrides the AMODE, reflected in the text file 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 16Mb, 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 16Mb
  - ORIGIN TRANS

If you specify neither AMODE nor RMODE, the AMODE is determined by the AMODE defined in the text file 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.

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 where the loaded program is to reside.

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 file ESD processing for this load.

Note: An AMODE value specified without an RMODE option defaults to RMODE 24 for the module.

If specified, RMODE will cause the file to load above or below the 16MB line of a virtual machine 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 file ESD for the entry point. The valid RMODE values and their meanings are:

24
The load resides below the 16MB line, overriding the RMODE definitions encountered on the text file ESD records during this load. An RMODE 24 definition is propagated to the GENMOD process.

ANY
The load resides above the 16MB line, overriding the RMODE definitions encountered on the text file ESD records during this load. 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.

hexloc
The program is loaded 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.

TRANS
The program is loaded into the CMS nucleus transient area.

For more information about using the ORIGIN option, see usage note "1" on page 475.

NOHOBSET
specifies the high-order bit of all V-type constants (VCONs) will be left unchanged. This is the default.

HOBSET
specifies the high-order bit is to be turned on for V-type constants (VCONs) of SD (CSECTs) and LD (ENTRYs) types that have AMODE ANY/31 addresses.

HOBSETSD
specifies the high-order bit is to be turned on for V-type constants (VCONs) of SD (CSECTs) type that have AMODE ANY/31 addresses.

FILETYPE *ft*
specifies the file type of the file(s) to be searched for and loaded. This file type is not used during text library searches (LIBE option).

UNDEF
displays undefined entry point names at the virtual console and prints them in the load map file (unless the NOMAP option is specified). The default is UNDEF.

NOUNDEF
suppresses the display and printing of undefined entry point names.

VER
writes Verify (VER) statement images in the load map file. The default is VER.

NOVER
suppresses the writing of Verify (VER) statements in the load map file.

Usage Notes

1. More information about using the ORIGIN option:

   The RMODE that results from using the ORIGIN option propagates to the start of the GENMOD process:

   - ORIGIN *xxxxxxxx*
     - If *xxxxxxxx* is below 16MB, the result is RMODE 24.
     - If *xxxxxxxx* is above 16MB, the result is RMODE ANY.
   - ORIGIN TRANS

     The result is RMODE 24.

Important
If you do not specify ORIGIN, RMODE, or AMODE, loading begins at the storage area defined by the SET LOADAREA command. For more information, see "SET LOADAREA" on page 961.

   You may issue the LOAD command with various ORIGIN/RMODE and AMODE options:

   ```
   LOAD pgmA pgmB ...
   LOAD pgmA pgmB ...(RMODE xxx AMODE xxx
   LOAD pgmA pgmB ...(RMODE xxx
   LOAD pgmA pgmB ...(ORIGIN xxxx AMODE xxx
   LOAD pgmA pgmB ...(ORIGIN xxxx
   LOAD pgmA pgmB ...(AMODE xxx
   ```

Table 20 on page 476 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 or INCLUDE commands.

Table 20. LOAD Process, RMODE and AMODE Determination, Load Residency

LOAD command options		On text file ESD record				RMODE / AMODE at end of load process	Residency of loaded programs[1]
		Load CSECT		Subsequent CSECT			
ORIGIN / RMODE	AMODE	RMODE	AMODE	RMODE	AMODE		
>16MB / ANY	24	n/a	n/a	n/a	n/a	993E 377E	993E 377E
>16MB / ANY	31	n/a	n/a	n/a	n/a	ANY/31	>16MB
>16MB / ANY	ANY	n/a	n/a	n/a	n/a	ANY/31 [2]ANY/ANY	<16MB
<16MB / 24	24	n/a	n/a	n/a	n/a	24/24	<16MB
<16MB / 24	31	n/a	n/a	n/a	n/a	24/31	<16MB
<16MB / 24	ANY	n/a	n/a	n/a	n/a	24/ANY	<16MB
>16MB / ANY		n/a	n/a	n/a	n/a	ANY/31	>16MB
<16MB / 24		n/a	24	n/a	n/a	24/24	<16MB
<16MB / 24		n/a	31	n/a	n/a	24/31	<16MB
<16MB / 24		n/a	ANY	n/a	n/a	24/ANY	<16MB
<16MB / 24		n/a	n/a	n/a	[3]	24/[3]	<16MB
	24	n/a	n/a	n/a	n/a	24/24	<16MB
	31	n/a	n/a	n/a	n/a	24/31	<16MB
	ANY	n/a	n/a	n/a	n/a	24/ANY	<16MB
		[4]	[4]	n/a	n/a	24/24	<16MB
		24	24	n/a	n/a	24/24	<16MB
		24	31	n/a	n/a	24/31	<16MB
		24	ANY	n/a	n/a	24/ANY	<16MB
		ANY	31	24	n/a	994W 24/31	994W <16MB
		ANY	31	24	[3]	994W 24/[3]	994W <16MB
		ANY	31	ANY	n/a	ANY/31	>16MB
		ANY	ANY	ANY	n/a	ANY/31 [2]ANY/ANY	>16MB <16MB
		ANY	n/a	ANY	[5]	ANY/31	>16MB

Table 20. LOAD Process, RMODE and AMODE Determination, Load Residency (continued)

LOAD command options		On text file ESD record				RMODE / AMODE at end of load process	Residency of loaded programs[1]
		Load CSECT		Subsequent CSECT			
ORIGIN / RMODE	AMODE	RMODE	AMODE	RMODE	AMODE		

Notes:

1

If a virtual machine has less than 16MB of storage, the loaded programs can reside only below 16MB. A specification of RMODE ANY, either on the LOAD command or text file ESD, would result in the LOAD residing below the 16MB line.

2

If a virtual machine has less than 16MB of storage, the loaded programs can reside only below 16MB. In this case, the loaded programs may execute in either 24-bit or 31-bit addressing mode. If you specified AMODE ANY on the LOAD command or defaulted to a text file 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 file ESD for that entry point.

4

If no RMODE and AMODE is specified (for example, 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. The combination causes an assembly error.

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 the DUP option is in effect. If a section contains an address constant that references a CSECT with a duplicate name that has not been loaded, that address constant may be resolved incorrectly.
3. The loader selects the entry point for the loaded program according to the following hierarchy:
   - Parameter list on the START command
   - Last RESET operand in a LOAD or INCLUDE command
   - Last ENTRY statement in the input
   - Last LDT statement in the input
   - First assembler- or compiler-produced END statement that specifies an entry point if no ENTRY statement is in the input
   - 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 load map will not contain the correct cumulative length values.
6. Common sections and pseudo-register cumulative length fields 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. For more information on the loader search order, see Figure 17 on page 478. 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. If you use any of the following nondefault options on the LOAD command, they remain in effect until you explicitly reset them to the defaults on a subsequent INCLUDE command (or until you issue another LOAD command, which resets everything):

   RLDSAVE, RMODE, HIST, NCHIST, FILETYPE, NOUNDEF, FULLMap

These options are not affected by the SAME option on the INCLUDE command.

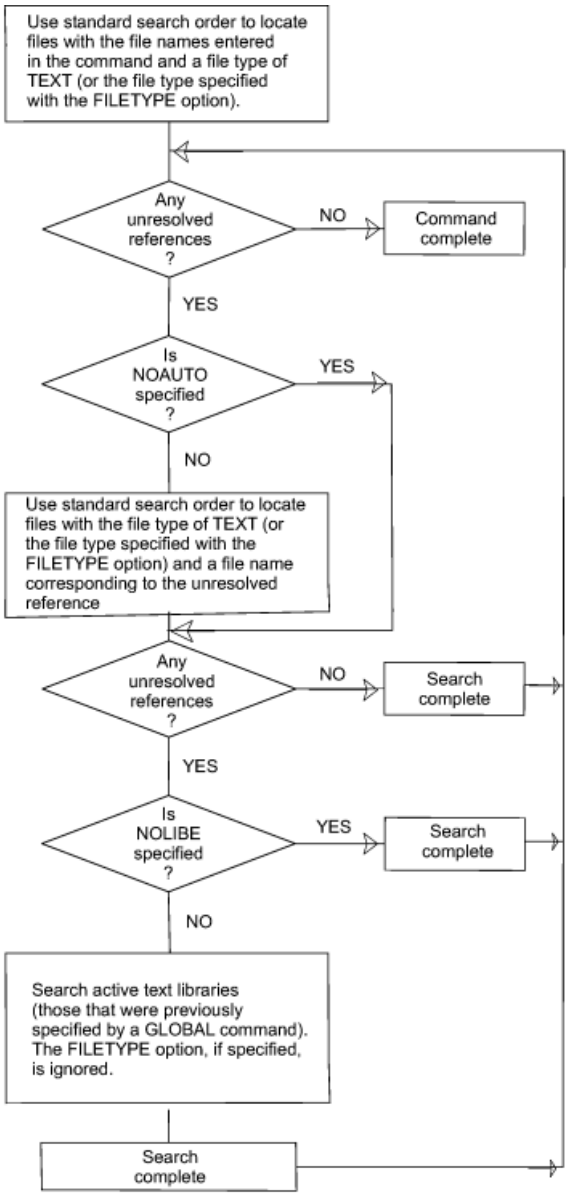


Figure 17. Loader Search Order

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, RMODE, nor AMODE on the LOAD command, the setting of the SET LOADAREA command determines where loading begins. If you did *not* issue SET LOADAREA 20000, the LOAD will start loading above or below the 16MB line based on the RMODE definition of the first text file ESD record encountered.
    - If a more restrictive RMODE is encountered, RMODE 24 after RMODE ANY was the first detected, the loader will stop loading above 16MB and will start loading below 16MB, unless no more storage is available to contain the load. If a reload occurs, you will receive message DMS994W.
    - When you specify the RLDSAVE option, the amount of relocation information the CMS loader can save is dependent on the available free storage in your virtual machine.
    - 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.

- 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.
- If you specify your first CSECT with a length of zero(0), the address in the load map will be specified as asterisks. These asterisks will represent the same location as the first valid address specified on the map after the LOAD has completed. To determine the actual address, use the:
  - Address produced by the map
  - PROGMAP command
- If the NOUNDEF option is used on the LOAD command, the UNDEF option can be used on the last INCLUDE command to display any entry points still unresolved. If any are found, the return code from INCLUDE is 4; if none are found, the return code is 0 (assuming no other errors).
- When using the FILETYPE option to specify the text file type, you should also use the NOAUTO option to suppress the automatic searching for text files, unless *all* the files to be searched for and loaded have the same file type.
- If the program you load requires storage currently held by another program, the other program is deleted, whether you specify the PRES option. Use the SET LOADAREA command to affect the program life of a nonrelocatable program. For more information, see *z/VM: CMS Application Development Guide for Assembler*.

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 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 21 on page 479. 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.

Table 21. ENTRY Statement Format

Column	Contents
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 22 on page 479. 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.

Table 22. LIBRARY Statement Format

Column	Contents
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 23 on page 480.

Table 23. LDT Statement Format

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 24 on page 480.

An ICS statement must be placed at the front of the file or text file.

Table 24. ICS Statement Format

Column	Contents
1	X'02' (12-2-9 punch). Identifies this as a loader control statement.
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 25 on page 481.

It sets the location counter in one of the following four 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 17-24.
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 17-24, 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.
4. Setting the location counter lower than its current value will generate a "not valid card" message in the load map.

Note: Be careful when using absolute values on a virtual machine with LOADAREA set to RESPECT.

Table 25. SLC Statement Format

Column	Contents
1	X'02' (12-2-9 punch). Identifies this as a loader control statement.
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 17-24. It must be right-justified in these columns, with unused leading columns filled with blanks or zeros.
13-16	Blank.
17-24	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.
25-72	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.

Verify (VER) Statement

A VER statement compares hexadecimal data contained in the statement with data at a specified location in virtual storage.

The format of a VER statement is shown in Table 26 on page 481. The data in columns 17-70 (excluding commas) is compared with what has already been loaded into virtual storage, beginning at the address specified in columns 5-12. VER statements are placed in the file following the TEXT statements and before the first RLD (relocatable dictionary) statement, if present. If Replace (REP) statements are used, the VER statements are placed immediately before the corresponding REP statements.

Note: If the verify function fails due to a not valid ESID or storage address, the loader terminates. If the verification is not valid, the loader completes its processing with a return code of 4.

Table 26. VER Statement Format

Column	Contents
1	X'02' (12-2-9 punch). Identifies this as a loader control statement.
2-4	VER - identifies the type of load statement.
5-12	Hexadecimal starting address of the area to be verified 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.

Table 26. VER Statement Format (continued)

Column	Contents
15-16	ESID (External Symbol Identification) — the hexadecimal number assigned to the control section in which the verification is to be made. The LISTING file produced by the compiler or assembler indicates this number.
17-70	A maximum of 11 four-digit hexadecimal fields, separated by commas, each verifying one previously loaded halfword (2 bytes). The last field must not be followed by a comma.
71-72	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 27 on page 482.

The data in columns 17-70 (excluding the commas) replaces what has already been loaded into virtual storage 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.

Table 27. REP Statement Format

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.
17-70	A maximum of 11 four-digit hexadecimal fields, separated by commas, each replacing one previously loaded halfword (2 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. It can be used anywhere in a text deck and causes the next CSECT to begin on a page boundary. The SPB statement updates the location counter even if it already points to a page boundary.

The SPB statement has the format shown in Table 28 on page 483.

This statement can be placed before, between, or after object modules or other control statements.

Table 28. SPB Statement Format

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

```
DMSLIO740I  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 in effect with LOAD and a not valid statement was found. The message and the contents of the not valid statement (xxx...xxx) are listed in the load map file. The statement is not valid, therefore, it is ignored and loading continues.

Load Maps

When the MAP or FULLMAP option is specified, a load map is written on your disk or directory accessed as A. If MAP is specified, the load map is named "LOAD MAP A5". If FULLMAP is specified, the load map is named "*fn* LOADMAP A5", where *fn* is the first file name specified on the LOAD command. If a file by that name already exists, it is replaced. If NOMAP is specified, a previously existing load map file on your disk accessed as A is erased and no new load map is created. If your file mode A is a directory, the existing load map file will not be changed to preserve the SFS attributes of this file. Unless you do specify NOMAP, you must have a read/write disk or directory accessed as A when you issue LOAD to contain the new load map created.

A load map created using MAP is a file containing the location of control sections, entry points of files loaded into storage, and the AMODE and RMODE information specified on the text file 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. It is a fixed format file of length 100 and appears as follows:

```
PROG1    SD 00020000          RMODE 24  AMODE 24
PROG2    SD 00020318          RMODE ANY AMODE 31
PROG3    SD 00020B48          RMODE 24  AMODE ANY
```

If the FULLMAP option is specified, in addition to the information created by using the MAP option, a variable format file is generated (up to 100 characters) including CSECT length, file name, file type, file mode, and the time stamp of the file of each ESD record. It appears as in this figure (without the title line):

```
Name     Ty  Origin    Length                          Fn       Ft      Fm  Member     Timestamp
PROG1    SD  00020000  00000318  RMODE 24  AMODE 24  PROG1    TEXT    A1             1/11/91  14:37:41
PROG2    SD  00020318  00000830  RMODE ANY AMODE 31  PROG2    TEXT    A1             1/11/91  14:37:50
PROG3    SD  00020B48  00000210  RMODE 24  AMODE ANY LIB1     TXTLIB  A1  PROG3     12/26/90  15:25:30
```

Figure 18. Information Provided by the FULLMAP Option

The MAP and FULLMAP options may not be overridden by a subsequent INCLUDE command. For example, if FULLMap is specified on LOAD and MAP is specified on a subsequent INCLUDE, the format of the load map will reflect the FULLMAP option.

You can use the DEFAULTS command to set up MAP|FULLMAP|NOMAP and NOHOBSET|HOBSET|HOBSETSD options for the LOAD command, or override the default option. However, the option you specify when entering the LOAD command overrides the one specified in the DEFAULTS command. This allows you to customize the default for the LOAD command, yet override it when you desire. For more information, see "DEFAULTS" on page 153.

If you specify the TYPE option, you can read the information in the load map file and on your terminal.

If a not valid 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.

Examples

1. ```
   DEFAULTS SET LOAD NOMAP
   LOAD DMSCSL (NOMAP
   ```

   Result: LOAD MAP A5 was erased and no new map file was created.

2. ```
   DEFAULTS SET LOAD MAP
   DEFAULTS SET INCLUDE FULLMAP
   LOAD TXT1 (NOMAP
   INCLUDE TXT2 (NOMAP
   INCLUDE TXT3 (NOMAP
   ```

   Result: LOAD MAP A5, TXT2 LOADMAP A5 AND TXT3 LOADMAP A5 were erased and no new map file was created.

3. ```
   DEFAULTS SET LOAD FULLMAP
   DEFAULTS SET INCLUDE FULLMAP
   LOAD TXT1 (NOMAP
   INCLUDE TXT2 (NOMAP
   INCLUDE TXT3 (NOMAP
   ```

   Result: TXT1 LOADMAP A5, TXT2 LOADMAP A5 AND TXT3 LOADMAP A5 were erased and no new map file was created.

4. ```
   DEFAULTS SET LOAD FULLMAP
   DEFAULTS SET INCLUDE MAP
   LOAD TXT1 (NOMAP
   INCLUDE TXT2 (NOMAP
   INCLUDE TXT3 (NOMAP
   ```

   Results: TXT1 LOADMAP A5 AND LOAD MAP A5 are erased and no new map file is created.

5. ```
   DEFAULTS SET LOAD NOMAP
   LOAD DMSCSL (MAP
   ```

   Results: the old LOAD MAP A5 is replaced by a new LOAD MAP A5.

6. ```
   DEFAULTS SET LOAD MAP
   LOAD DMSCSL (MAP
   ```

   Results: the old LOAD MAP A5 is replaced by a new LOAD MAP A5.

7. ```
   DEFAULTS SET LOAD FULLMAP
   LOAD DMSCSL (MAP
   ```

   Results: the old LOAD MAP A5 is replaced by a new LOAD MAP A5.

8. ```
   DEFAULTS SET LOAD NOMAP
   LOAD DMSCSL (FULLMAP
   ```

   Results: the old DMSCSL LOADMAP A5 is replaced by a new DMSCSL LOADMAP A5.

9. ```
   DEFAULTS SET LOAD MAP
   LOAD DMSCSL (FULLMAP
   ```

   Results: the old DMSCSL LOADMAP A5 is replaced by a new DMSCSL LOADMAP A5.

10. ```
    DEFAULTS SET LOAD FULLMAP
    LOAD DMSCSL (FULLMAP
    ```

    Results: the old DMSCSL LOADMAP A5 is replaced by a new DMSCSL LOADMAP A5.

Messages and Return Codes

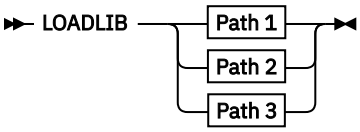
- DMS001E No filename specified [RC=24]
- DMS002I File *fn* TXTLIB not found
- DMS002E File[(s)] *fn* TXTLIB not found [RC=28]
- DMS003E Invalid option: *option* [RC=24]
- DMS005E No *option* specified [RC=24]
- DMS021E Entry point *name* not found [RC=40]
- DMS029E Invalid parameter *parameter* in the option *option* field. [RC=24]
- DMS055E No entry point defined [RC=40]
- DMS056E File *fn ft [fm]* contains invalid [RLD] record formats [in *entryname*] [RC=32]
- DMS099E CMS/DOS environment not active [RC=40]
- DMS104S Error *nn* reading file *fn ft fm* from disk [RC=31 | 55 | 70 | 76 | 99 | 100]
- DMS105S Error *nn* writing file *fn ft fm* on disk [RC=31 | 55 | 70 | 76 | 99 | 100]
- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS116S Loader table overflow [RC=104]
- DMS168S Pseudo register table overflow [RC=104]
- DMS169S ESDID table overflow [RC=104]
- DMS201W The following names are undefined: *namelist* [RC=4]
- DMS202W Duplicate identifier *identifier* [RC=4]
- DMS203W SET LOCATION COUNTER *name* undefined [RC=4]
- DMS206W Pseudo register alignment error [RC=4]
- DMS257T Internal system error at address *addr* (offset *offset*)
- DMS377E AMODE of 24 specified with RMODE of ANY, LOAD failed. [RC=68]
- DMS379E INCLUDE address at or above 16MB conflicts with LOAD address below 16MB, INCLUDE failed. [RC=88]
- DMS379E INCLUDE address below 16MB conflicts with LOAD address at or above 16MB, INCLUDE failed. [RC=88]
- DMS380E Storage at origin *addr* in use, *file* not loaded. [RC=104]
- DMS381E Insufficient storage available below 16MB to load *file*. [RC=88]

- DMS623S Module cannot be loaded at location *location*—this area is available for system use only [RC=88]
- DMS907T I/O error on file *fn ft fm*
- DMS943E Invalid AMODE *mode* specified [RC=24]
- DMS993E AMODE of 24 cannot be specified with ORIGIN address greater than 16MB, LOAD failed. [RC=68]
- DMS994W Restrictive RMODE encountered in CSECT *cname*. LOAD continues below 16MB.
- DMS997E The specified ORIGIN address is outside the virtual machine size, LOAD|INCLUDE failed. [RC=64]
- DMS1144E Implicit rollback occurred for work unit *workunitid* [RC=31]
- DMS1220E ORIGIN is invalid when specified with RMODE. [RC=68]
- DMS1252T Rollback unsuccessful for file pool *filepoolid*

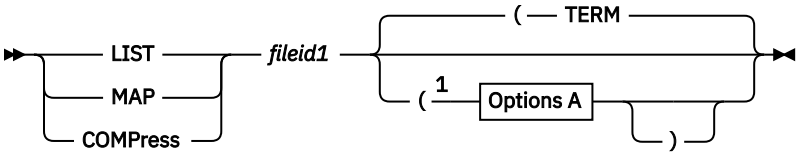
Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in the Shared File System	“File Pool Server Messages” on page 1414
Errors in using a file	“File Error Messages” on page 1418

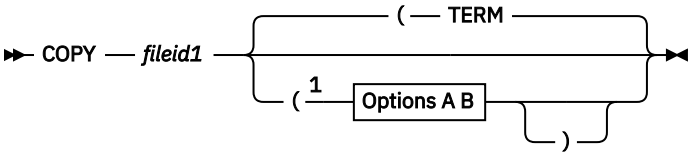
LOADLIB



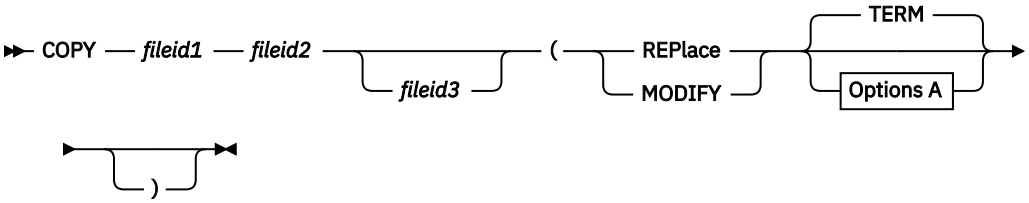
Path 1



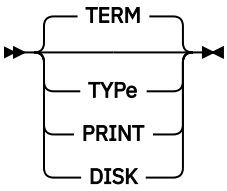
Path 2



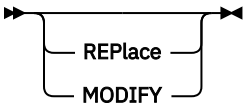
Path 3



Options A



Options B



Notes:

[1] You can enter Options in any order between the parentheses.

Authorization

General User

Purpose

Use the LOADLIB command to list, map, copy, or compress a CMS LOADLIB. CMS LOADLIBs can be merged, and specified members can optionally be selected or excluded during the merge.

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 this is not supported.

MAP
has the same function as the LIST operand.

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

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: 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:

```
membername(R
```

or

```
membername(R)
```

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 COPY function before specifying new file identifiers in subsequent LOADLIB commands.

Note: You may specify the LOADLIB function (LIST, MAP, 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. (You must use the full function name in the SYSIN data set, not the abbreviation. You must also use the full name of any options you have specified in the command line.) 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 with 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 directly specified as input; it must be related to a CMS file ID using the FILEDEF command.

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*.

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.

Note: The file containing the SYSIN control statements must be a Fixed 80 (RECFM=F, LRECL=80) file.

Options

Options Entered in the Command Line

TERM
directs printer output to the terminal. The default is TERM.

TYPe
directs printer output to the terminal. (This option is the same function as the TERM option.)

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 A1.

REPlace
replaces all existing members of a data set and adds new members. To replace only selected members of a data set, see the Select operand.

MODIFY
does not replace existing members of a data set; adds new members.

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 you choose 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.

Examples

These are two examples of using FILEDEF and LOADLIB:

1. In this case, the function (COPY) is specified in the command line, so the abbreviation REP can be used for the REPLACE option.

```
filedef sysut1 disk lib1 loadlib a
filedef sysut2 disk lib2 loadlib a
filedef sysin disk control file a
loadlib copy (rep term
```

   Where in CONTROL FILE A you have:

```
SELECT
MEMB1
MEMB2
```

2. In this case, the function (COPY) is specified in the control file. The full option name, REPLACE, must therefore be used in the command line.

```
filedef sysut1 disk lib1 loadlib a
filedef sysut2 disk lib2 loadlib a
filedef sysin control file a
loadlib (replace term
```

   Where in CONTROL FILE A you have:

```
COPY
SELECT
MEMB1
MEMB2
```

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

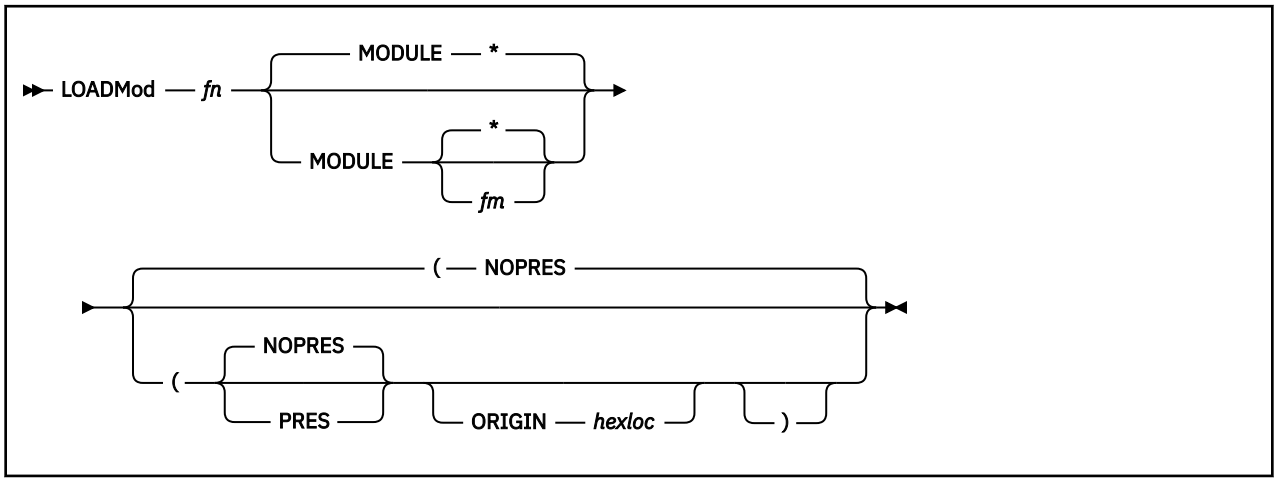
Messages and Return Codes

- DMS003E Invalid option: *option*
- DMS014E Invalid function *function* [RC=24]
- DMS024E File *fn ft fm* already exists [RC=28]
- DMS032E Invalid filetype *ft* [RC=24]
- DMS037E Output filemode *mode* is accessed as read/only [RC=36]
- DMS039E No entries in library *fn ft fm* [RC=32]
- DMS042E No fileid(s) specified [RC=24]
- DMS047E No function specified [RC=24]
- DMS048E Invalid file mode *fm* [RC=24]
- DMS054E Incomplete fileid specified [RC=24]
- DMS062E Invalid character *char* in fileid *fn ft* [RC=20]
- DMS065E *option* option specified twice [RC=24]
- DMS066E *option1* and *option2* are conflicting options [RC=24]
- DMS069E Filemode *mode* is not accessed [RC=36]
- DMS073E Unable to open file *ddname* [RC=28]
- DMS188W SYSUT2 header is invalid because of blocksize incompatibility; user action required [RC=4]
- DMS189E The LIST function of the LOADLIB command does not support concatenated SYSUT1 [RC=24]
- DMS901T Unexpected error at *vstor1*: plist *function fn ft fm* at *vstor2*, base *vstor3*, rc=*nn* [RC=31 | 55 | 70 | 76 | 99 | 256]
- DMS907T I/O error on file *fn ft fm* [RC=256]
- DMS1144E Implicit rollback occurred for work unit *workunitid* [RC=31]
- DMS1252T Rollback unsuccessful for file pool *filepoolid*

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in the Shared File System	“File Pool Server Messages” on page 1414
Errors in using a file	“File Error Messages” on page 1418
Errors in copying a file	“Messages and Return Codes” on page 90
Errors in command syntax	“Command Syntax Error Messages” on page 1411

LOADMOD



Authorization

General User

Purpose

Use the LOADMOD command to load a CMS module into storage.

Note: If the module file was not created using the GENMOD command with the MAP option, the loaded module will produce unpredictable results.

CMS will respect the AMODE and RMODE attributes established by the GENMOD process. When LOADMOD executes, the RMODE attribute assigned to the module at generation time determines whether that file resides above or below the 16MB line. A subsequent START command causes the module to execute with the AMODE established by the GENMOD process. To view the AMODE and RMODE attributes prior to executing the module with the START command, issue the PROGMAP command.

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 are searched for the file.

Options

NOPRES
instructs CMS to release the storage 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 previously loaded by a LOAD, INCLUDE, or LOADMOD command. The default is NOPRES.

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.

ORIGIN
hexloc
instructs CMS to load the specified module into storage at the hexadecimal address specified by *hexloc*.

Usage Notes

1. You can use the LOADMOD command when you want to debug a CMS MODULE file produced by the GENMOD command with the MAP option. Because 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:

```
loadmod prog1
progmap
cp per instruct range xxxxxx
start
```

   Another procedure you can use with modules generated by either the GENMOD command with the NOMAP option or by the BIND command is as follows:

```
nucxload prog1
nucxmap prog1
cp trace instruct range xxxxxx
prog1
…
nucxdrop prog1
```

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 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 AMODE or RMODE attributes if they:
   - Were created by the LOAD command with the ORIGIN TRANS option
   - Used the SYSTEM option from GENMOD

   The default is AMODE 24 and RMODE 24. The PROGMAP command does not display the attributes of transient area modules.
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, and issue message DMS380E. There are three possible solutions to this situation:
   a. Regenerate the nonrelocatable program so it will be relocatable. This is the simplest method.
   b. Use the ORIGIN option on the LOAD command to change the address where the nonrelocatable program gets loaded to a lower address in storage. This way, the program will reside at an address to be allocated at a later time.
   c. 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 virtual machine with less than 16MB of storage.
8. The ORIGIN option is valid for both relocatable and nonrelocatable modules. However, specifying an ORIGIN address for a nonrelocatable module different from the address where the module was generated may cause unpredictable results when the module is executed. To determine whether a module is relocatable or nonrelocatable and the address where the nonrelocatable module was

generated, issue the LOADMOD command without the ORIGIN option, followed by the PROGMAP command. PROGMAP displays the attributes of any non-transient area modules loaded into storage.

9. An ORIGIN address located in the transient area (X'E000' to X'10000') will cause the DMS380E message to be issued and the module will not be loaded.
10. Specifying the ORIGIN option for a module generated in the transient area will cause the ORIGIN option to be ignored. The module will always load in the transient area.
11. Specifying an ORIGIN address not aligned on a doubleword boundary will cause the module to be loaded at the previous doubleword boundary. For example, issuing `LOADMOD MYPROG (ORIGIN 30003` will cause MYPROG to be loaded at 30000. This address will be reflected as the origin address when the PROGMAP command is issued.
12. If the module cannot be loaded at the address specified on the ORIGIN option, either message DMS380E or message DMS997E will be issued and the module will not be loaded.

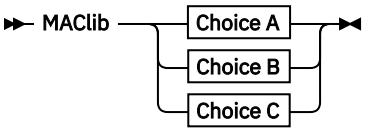
Messages and Return Codes

- DMS001E No filename specified [RC=24]
- DMS002E File[(s)] [fn [ft [fm]]] not found [RC=28]
- DMS003E Invalid option: *option* [RC=24]
- DMS005E No ORIGIN specified [RC=24]
- DMS018E No load map available [RC=40]
- DMS029E Invalid parameter *parameter* in the option ORIGIN field [RC=24]
- DMS032E Invalid filetype *ft* [RC=24]
- DMS037E Filemode *mode*[(*vdev*)] is accessed as read/only [RC=36]
- DMS069E Filemode *mode* not accessed [RC=36]
- DMS070E Invalid parameter *parameter* [RC=24]
- DMS104S Error *nn* reading file *fn ft fm* from disk [RC=31 | 55 | 70 | 76 | 99 | 100]
- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS114E *fn ft fm* not loaded; CMS/DOS environment [not] active [RC=40 or -0005]
- DMS116S Loader table overflow [RC=104]
- DMS257T Internal system error at address *addr* (offset *offset*)
- DMS380E Storage at origin *addr* in use, *fn* not loaded [RC=104]
- DMS639E Error in routine *routine*, return code was *nnn* [RC=*nnn*]
- DMS945E AMODE/RMODE values conflict. *file* not loaded | generated [RC=68]
- DMS988E Module *fn* cannot execute in {XA | XC} architecture [RC=64]
- DMS997E The specified ORIGIN was outside of the virtual machine size, LOADMOD ORIGIN failed [RC=46]
- DMS1144E Implicit rollback occurred for work unit *workunitid* [RC=31]
- DMS1252T Rollback unsuccessful for file pool *filepoolid*

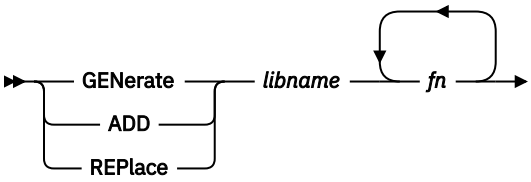
Additional system messages may be issued by this command. The reasons for these messages and their location are:

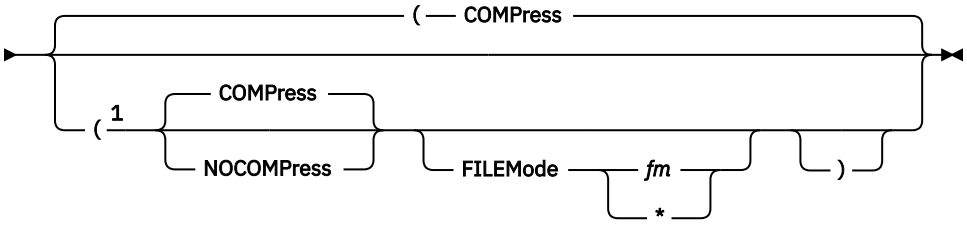
Reason	Location
Errors in Shared File System	“File Pool Server Messages” on page 1414
Errors in using a file	“File Error Messages” on page 1418

MACLIB

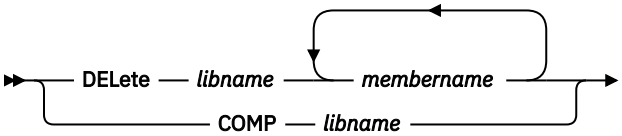


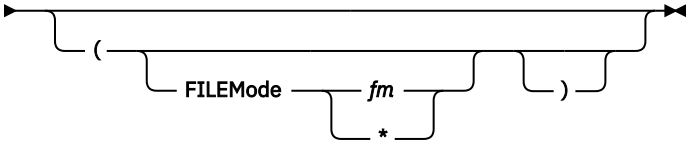
Choice A



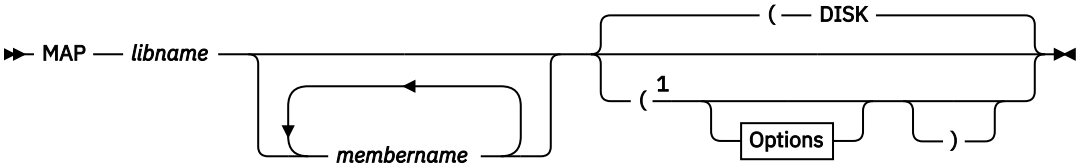


Choice B





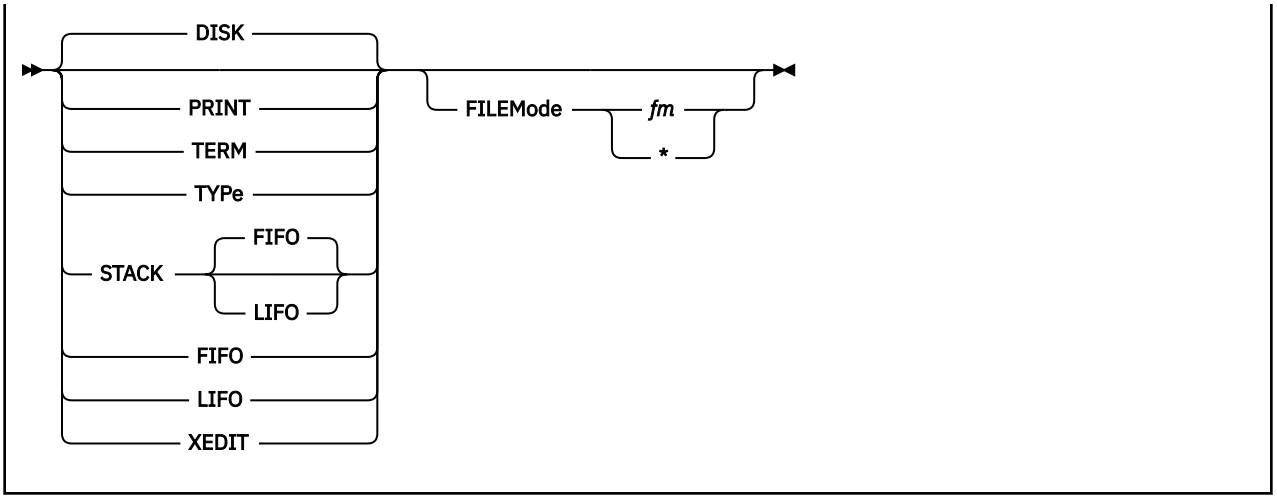
Choice C



Notes:

[1] You can enter Options in any order between the parentheses.

Options



Authorization

General User

Purpose

Use the MACLIB command to create and modify CMS macro libraries.

Operands

GENerate
generates a CMS macro library.

ADD
adds members to an existing macro library. No checking is done for duplicate names, entry points, or CSECTS.

REPlace
replaces existing members in a macro library.

DELete
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.

fn
represents the name(s) 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.

membername
represents the name(s) of the macros that exist in a macro library.

Options

COMPress
specifies comment lines in the source file that start in column 1 and begin with .*! should be removed from the macro source file when the macro is included in a macro library. This is the default.

Note: You identify those internal macro comments that are extraneous (for example, a prolog) by putting an exclamation point (!) after the .* that begins the comment.

NOCOMPress
specifies comments are not to be removed from the macro source when the macro is included in a macro library.

The following options specify where the output of the MAP function is sent. If you specify more than one option, CMS uses the last option specified.

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, the default is DISK.

PRINT
spools the MACLIB map to your virtual printer.

TERM
displays the MAP output at the terminal.

TYPe
Synonym for TERM option.

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. 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.

FILEMode *fm*
specifies the file mode of the macro library you are creating or changing. If specified as FILEMODE *, all accessed minidisks and directories are searched.

If creating a new file with the GENERATE option and this option is either not specified or specified as *, the file is created on file mode A. If A is not accessed, you will receive error message DMSOPN069E with a return code of 36. If this option is not specified or specified as FILEMODE * and you are changing an existing file, all accessed files and directories are searched for the file.

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 macrodefinitions 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 records not valid between members, the MACLIB command displays an error message DMSLBM056E and ends. Any members read before the not valid 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 component are: DMSGPI, DMSOM, OSMACRO, OSPSI, OSMACRO1, MVSXA, and OSVSAM.
6. MACLIB COMP writes a temporary work file named DMSLBM CMSUT1 on your disk or directory accessed as A, so you must have a disk or directory accessed as A accessed read/write when you use MACLIB COMP.
7. The PRINT, TYPE, and TERM options erase the old MAP file, if one exists.
8. If MACRO and COPY files exist which have the same file name and search priority, MACLIB REPLACE, ADD, and GENERATE functions use the MACRO version first.
9. If any MACRO or COPY file contains records of variable record format, have a record length not equal to 80, or have 0 records, between members, the MACLIB command displays error message DMSLBM056E and ends.
10. A valid format maclib is a fixed record format with a record length equal to 80. If MACLIB encounters a not valid maclib, MACLIB (except MACLIB GENerate) will stop processing and issue an error message.
11. A return code of 4 indicates 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 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, and other applications may produce unpredictable results.

Responses

When you enter the MACLIB MAP command with the TERM or TYPE 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

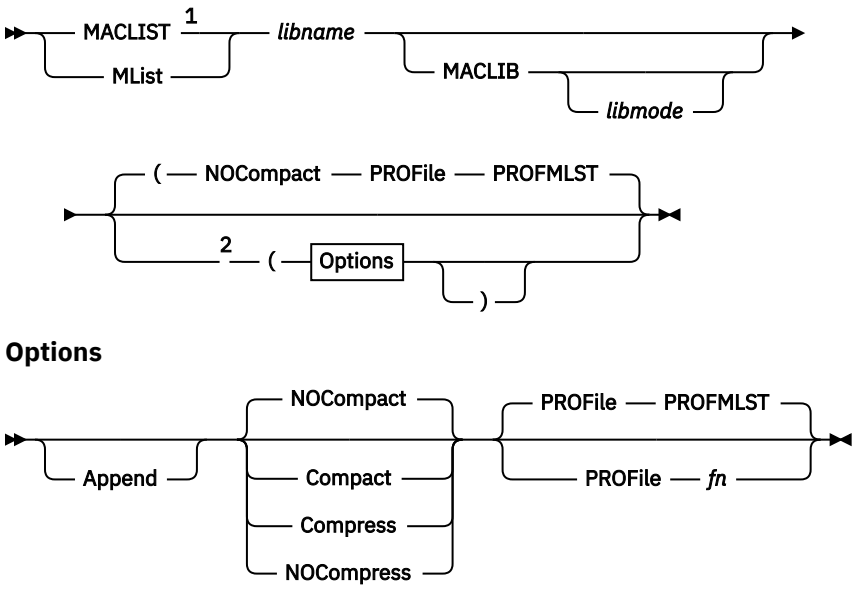
- DMS002W File *fn [ft] [fm]* not found [RC=4 or 28]
- DMS013W Member *membername* not found in library *libname* [RC=4]

- DMS037E Filemode *mode*[(*vdev*)] is accessed as read/only [RC=36]
- DMS056E File *fn ft [fm]* contains invalid record formats [RC=32]
- DMS069E Filemode *mode* not accessed [RC=36]
- DMS104S Error *nn* reading file *fn ft fm* from disk or directory [RC=31 | 55 | 70 | 76 | 99 | 100]
- DMS105S Error *nn* writing file *fn ft fm* on disk or directory [RC=31 | 55 | 70 | 76 | 99 | 100]
- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS157S MACLIB limit exceeded[, last member added was *membername*] [RC=88]
- DMS167S Previous MACLIB function not finished [RC=88]
- DMS213W Library *fn* MACLIB not created [RC=4]
- DMS213W Library *fn* MACLIB not created, or erased if empty [RC=4]
- DMS213W Library *fn* MACLIB has no members [RC=4]
- DMS688E XEDIT option only valid from XEDIT environment [RC=24]
- DMS689E File must be F-format 130 or V-format [RC=24]
- DMS907T 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 their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411
Errors in the Shared File System	“File Pool Server Messages” on page 1414
Errors in using a file	“File Error Messages” on page 1418
Errors in copying a file	“Messages and Return Codes” on page 90
Errors in erasing a file	“Messages and Return Codes” on page 217
Errors in renaming a file	“Messages and Return Codes” on page 826

MACLIST



Notes:

[1] Special commands that can be used in the MACLIST environment are EXECUTE and DISCARD.
[2] You can enter Options in any order between the parentheses.

Authorization

General User

Purpose

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 usually provided by the MACLIB command (with the MAP option) is displayed under the control of XEDIT. You can use XEDIT subcommands to manipulate the list itself.

Operands

libname
is the file name of the CMS macro library for which information is to be displayed.

MACLIB
is the file type of the CMS macro library for which information is to be displayed.

libmode
is the file mode of the CMS macro library for which information is to be displayed.

Options

If you specify more than one option, CMS uses the last option specified.

Append
specifies 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 the library is to be compacted upon completion of MACLIST using the MACLIB COMP command.

NOCompact
specifies the library is not to be compacted upon completion of MACLIST. This is the default.

Compress
specifies the library is to be compressed upon completion of the MACLIST command. (This option has the same function as the COMPACT option.)

NOCompress
specifies the library is not to be compressed upon completion of the MACLIST command. (This option has the same function as the NOCOMPACT option.)

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 Usage Note, “10” on page 503, "Default Key Settings".

Usage Notes

1. For more information on how to customize this command see Appendix A, “Customizing Profiles for CMS Productivity Aids,” on page 1419.
2. 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. For more information, see “EXECUTE” on page 1393. The DISCARD command allows you to erase the MACLIB members displayed by MACLIST. For more information, see “DISCARD” on page 1392.
3. Tailoring the MACLIST Command Options

   You can use the DEFAULTS command to set up options and 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. For more information, see “DEFAULTS” on page 153.
4. 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 “Examples” on page 504. 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, and so forth.

   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.
5. 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.
6. 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, and typing the command in the space provided to the left of the member name.

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 are finished typing the command, press Enter. If there is extra data on the line from a previous command, erase the rest of the line by pressing Erase EOF, or spacing over the rest of the line, before pressing Enter. For more information, see "EXECUTE" on page 1393. You may use the DELETE key to erase the rest of the line, but do not use it to erase only part of the rest of the line. For more information on how the command is interpreted, see "EXECUTE" on page 1393.

When you press Enter, 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. For more information, see "EXECUTE" on page 1393.

7. 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. For examples, see "Examples" on page 504.

   These symbols can be used:

   /
   means the *libname libtype libmode* (MEMBER *membername* that is displayed on the line.

   /l
   means the library file name displayed on the line.

   /t
   means the library file type displayed on the line.

   /m
   means the library file mode 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 file name followed by library file type.

   /lt
   means the library file name followed by library file type.

   /tl
   means the library file type followed by library file name.

   /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.

   Responses

8. Special Symbols Used Alone

   These special symbols can be typed alone on the lines of the MACLIST display. The meanings are:

   =
   specifies 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 members preceded by equal signs are discarded when the EXECUTE command is entered (from the command line or by pressing Enter).

   ?
   specifies display the last command executed. The command is displayed on the line in which the ? is entered.

   /
   specifies make this line the current line. (On the MACLIST screen, the current line is the first member name on the screen.)

9. If a member is a duplicate name and it is not the first one found in the macro library, you cannot issue any CMS commands, XEDIT subcommands, or any special symbols (=, ?, and /) for that member.
10. Default Key Settings

    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 these values:

Table 29. Default Key Settings Set by PROFMLST XEDIT

Key	Setting	Action
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).
PF1	Help	Display MACLIST command description.
PF2	Refresh	Update the list to indicate new members, deleted members, and so forth, using the same parameters as those specified when MACLIST was invoked.
PF3	Quit	Exit from MACLIST.
PF4	Sort(name)	Sorts by member name.
PF5	Sort(index)	Sorts by index, largest first.
PF6	Sort(size)	Sorts by size, largest first.
PF7	Backward	Scroll back one screen.
PF8	Forward	Scroll forward one screen.
PF9	Fl /n	Issue the command FILELIST /n * * at the cursor, so 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).
PF10	–	Not assigned.
PF11	XEDIT	Edit the member where the cursor is placed.
PF12	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 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).

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

```
exec maclist
```

12. MACLIST does not support an asterisk (*) for any of the input parameters, libname, libmode, or on the type field which must be MACLIB.
13. A valid format maclib is a fixed record format with a record length equal to 80. If MACLIB encounters a not valid maclib, MACLIB (except MACLIB GENerate) will stop processing and issue an error message.

Examples

The MACLIST screen in Figure 19 on page 504 was created by issuing the MACLIST command as follows:

```
maclist mylib
```

Note: The members are sorted alphabetically by member name. Members with the same name are then sorted by index number, the 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 19. 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 (FILEM fm
listfile /lt *	LISTFILE MYLIB MACLIB *
copyfile /ltm /l oldlib /m	COPYFILE MYLIB MACLIB fm MYLIB OLDLIB fm

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 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

- DMS002E File *fn* MACLIB * not found [RC=28]
- DMS032E Invalid filetype *ft* [RC=24]
- DMS213W Library *fn ft* has no members [RC=4]
- DMS651E APPEND must be issued from MACLIST [RC=40]
- DMS668E The APPEND option must be specified alone [RC=40]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

MAKEBUF

Authorization

General User

Purpose

Use the MAKEBUF command to create a new buffer within the program stack.

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 LINERD or RDTERM macro, the WAITRD function, the PULL or PARSE REXX instruction, or the &READ statement in EXEC or EXEC 2 to read lines from the buffers created by MAKEBUF. For more information, see *z/VM: CMS Macros and Functions Reference*.
3. When reading a line from the most recently created buffer, if this buffer is empty, it will be dropped (except for buffer 0) and the line will be read from the next buffer. This process will continue until a line is read. If there is no data to be read from the program stack, a line will be read from the terminal input buffer. If there is no data in the terminal input buffer, a line will be read from the virtual console. For more information on the program stack, see *z/VM: CMS Application Development Guide*.

MODMAP

Authorization

General User

Purpose

Use the MODMAP command to display the load map associated with the specified MODULE file.

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
  .           .
  .           .
  .           .
```

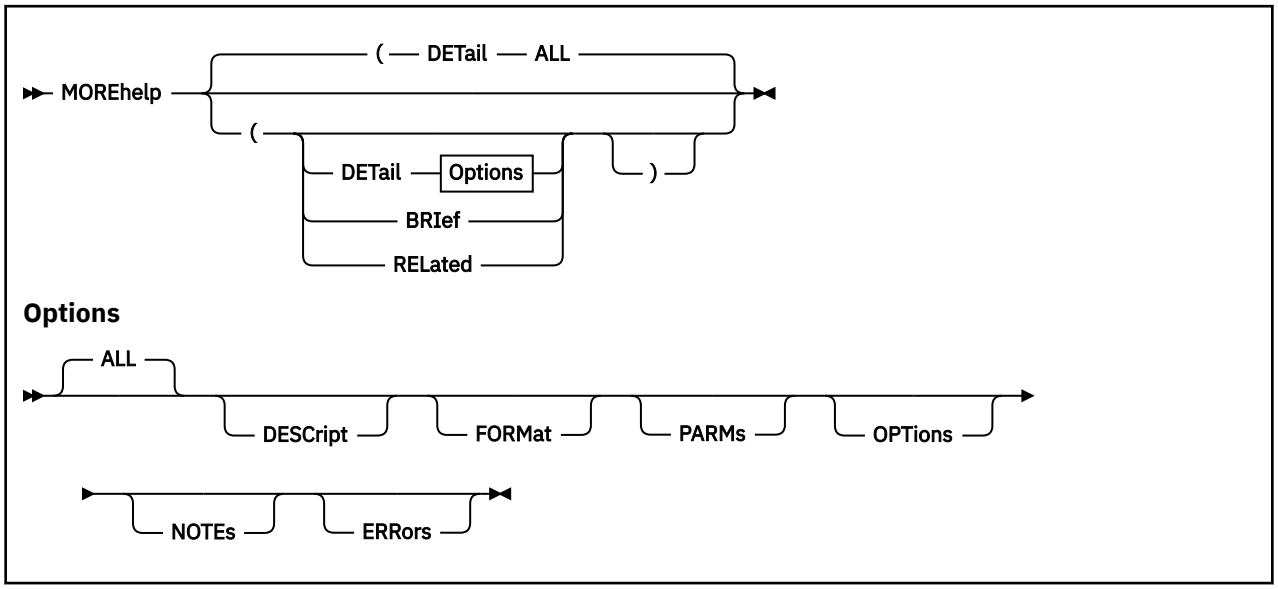
Messages and Return Codes

- DMS001E No filename specified [RC=24]
- DMS070E Invalid parameter *parameter* [RC=24]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

MOREHELP



Authorization

General User

Purpose

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 line mode terminal or if you cannot use a PF key to obtain DETAIL or RELATED information.

Options

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. The default is DETAIL ALL.

BRIef
displays the BRIEF layer of the HELP file. If the BRIEF layer is not available, you will see the next available layer of information.

RELated
displays the RELATED layer of the HELP file, if available.

Note: The following options can only 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. The default is ALL.

DESCript
displays the description information of the HELP file.

FORMat
displays the format information (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 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, 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 choose to tailor your own HELP files to include this option, you may do so. For more information, see "Tailoring the HELP Facility" in *z/VM: 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
```

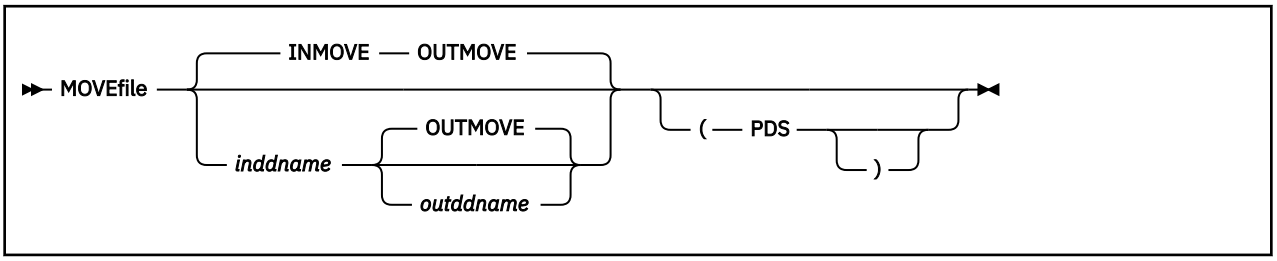
Messages and Return Codes

- DMS353E No previous HELP command has been entered. For more information on the MOREHELP command, enter, HELP MOREHELP [RC=4]
- DMS354E RELATED information is not available for the last HELP command entered. [RC=32]
- DMS639E Error in *routine* routine; return code was *xx* [RC=*xx*]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

MOVEFILE



Authorization

General User

Purpose

Use the MOVEFILE command to move data from any device supported by z/VM to any other device supported by z/VM.

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.

Options

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. If the input to MOVEFILE is a printer file spooled to the user's reader, the carriage control characters will not be available if the CC option was specified on the PRINT command which created the file. They will also not be available if the print file was created by an application program that uses carriage control characters.
2. 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, 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.

The file being moved is the *next* file in your 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 they are held. You can make a specific reader file the *next* file by ORDERing the reader prior to issuing the MOVEFILE command.

3. 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: The output ddname was not specified in full, so that CMS assigned the default file definition (FILE ddname).

4. You cannot copy VSAM data sets with the MOVEFILE command.
5. You can use the MOVEFILE command to move data contained in spanned records. If your FILEDEF command specifies the record format as DS, DBS, VS, or VBS, the logical record interface (LRI) will be used.
6. When using MOVEFILE on a file with spanned record formats, the default LRECL is 32756, and the record area is 32756+32.
7. MOVEFILE handles record format defaults for ANSI tapes in this manner:
   - On input, if RECFM is not specified or RECFM is specified as U or V, record format D is used.
   - On output, the record format of the input file is used. However, if you are writing to a non-ANSI tape or other devices and the input file's record format is D, record format V is used. If you are writing to an ANSI tape and the input file's record format is V, record format D is used.
8. To copy an entire partitioned data set into another partitioned data set, use the COPYFILE command. If an attempt is made to use MOVEFILE 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.

   Note: If MOVEFILE is used without the PDS option for a TXTLIB or a MACLIB, it will be treated as a partitioned data set.

9. 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 multivolume tape processing, DISP MOD is only valid with the tape currently mounted; no volume switching will be done.
10. 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.
11. 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 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.
12. For OS compatibility of the output tape labels, you must specify LRECL and BLOCK/BLKSIZE in your output FILEDEF.
13. Any attempt to move an empty OS data set that has not been closed will cause unpredictable results.

14. The MOVEFILE command does not support a record of zero bytes in a disk file.
15. The MOVEFILE command does not support multivolume tape processing for the 9346 cartridge tape unit.
16. The 9346 cartridge tape can only be written to at load point and at logical end of tape.
17. If the output device specified is a printer and the LRECL specified is larger than the blocksize being used, the length of the data which is moved is determined as follows: for an F or U record format, the length of the data being moved equals the blocksize; for a D or V record format, the length of the data being moved equals the blocksize minus 4.
18. If the output device specified is a tape, an EOF (end of file) tape mark will be written after the last output data record. The tape label type specified in the FILEDEF determines whether more than one tape mark is written signalling EOF or EOV (End Of Volume). Table 30 on page 512 lists what tape marks and labels are written:

Table 30. Tape Marks and Labels Written When Output is to Tape

Tape Label Type	Number of Tape Marks After Data	Trailer Label Types	Number of Tape Marks After Labels	CLOSE LEAVE Positioning
SL	1	EOF*x*	2	Between last two tape marks
SL	1	EOV*x*	1	Not applicable (new tape mounted)
SUL	1	EOF*x*	2	Between last two tape marks
SUL	1	EOV*x*, UTL*x*	1	Not applicable (new tape mounted)
AL	1	EOF*x*	2	Between last two tape marks
AL	1	EOV*x*	2	Not applicable (new tape mounted)
AUL	1	EOF*x*	2	Between last two tape marks
AUL	1	EOV*x*, UTL*x*	2	Not applicable (new tape mounted)
BLP	2	Not applicable	Not applicable	Between last two tape marks
NL	2	Not applicable	Not applicable	Between last two tape marks
LABOFF (default)	1	Not applicable	Not applicable	After tape mark

19. MOVEFILE can be used to process non-OS CMS type files with LRECL sizes up to 65535. However, you must take care when writing a CMS file type to a true OS type file, because the maximum nonspanned LRECL for OS fixed files is 32760. Normal OS variable files have a maximum LRECL of 32756. Only OS variable spanned files under Extended Logical Record Interface (XLRI) processing can range to an LRECL of 65535. Even then, you must allow for a 4-byte record length field as part of the total record data length. Therefore, the effective length of the actual data in the OS record is really 65531.

    Casual movements between CMS and true OS file types can cause data to be truncated with no notification when large LRECL values are involved. For example, a module file with the maximum CMS RECL of 65535 cannot be moved into an OS variable spanned format and then restored, because data will be truncated in the OS format.

    For a definition of OS type data, see "FILEDEF" on page 266. For more information on internal OS data formats and variable record length fields, see *z/VM: CMS Application Development Guide for Assembler*.

Default Device Attributes

If a record format (RECFM), block size (BLOCK or BLKSIZE), 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. For example, if a BLKSIZE is specified in the input file definition, MOVEFILE will truncate anything larger than what is specified, without warning, on the output tape.

- If the block size was not specified, the default is set according to Table 31 on page 513.
- If the logical record length is not specified,
  - For F or U record formats, the default logical record length is equal to the block size specified in Table 31 on page 513.
  - For D or V record formats, the logical record length equals the block size specified in Table 31 on page 513 minus 4.
  - For more information on the default block size and logical record length and interface (used with spanned records), see "FILEDEF" on page 266.

Table 31. Default Device Attributes for MOVEFILE Command

Device	Input *ddname*		Output *ddname*	
	RECFM	**Block size**	**RECFM**	**Block size**
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*	Block size of input *ddname*
ANSI Tape	D	2048	RECFM of input *ddname*	Block size of input *ddname*
Disk or Directory	RECFM of file	Block size of file	RECFM of input *ddname*	Block size of input *ddname*
Dummy	NA[1]	NA[2]	RECFM of input *ddname*	Block size of input *ddname*

Notes:

1. If the default record format and block size 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.

Conflicting Default Attributes May Arise

If the user does not specify file attributes completely using FILEDEF (RECFM, LRECL, and BLKSIZE), MOVEFILE may substitute or default the omitted attributes. These attributes can conflict with other attributes previously specified by the user. MOVEFILE obtains default information from the CMS file through an ESTATE. This information does not reflect true OS/SIMULATION attribute characteristics unless file mode type is 4.

Examples

```
"TEST FILE A1" has the following characteristics:

          RECFM   = V
          LRECL   = 19
```

Example 1

```
"FILEDEF IN DISK TEST FILE A1 (RECFM V LRECL 24 BLKSIZE 28"
"FILEDEF OUT TERM"
"MOVEFILE IN OUT"
```

This file definition will work properly because the LRECL is at least four bytes bigger than the actual CMS file, and the BLKSIZE is at least 4 bytes bigger than the LRECL. The four byte difference is due to the Record Descriptor Word and the Block Descriptor Word used for variable records in OS/SIMULATION. For more information on using OS/MVS Simulated data sets in CMS, see *z/VM: CMS Application Development Guide for Assembler*.

Example 2

```
"FILEDEF IN DISK TEST FILE A1 (RECFM V LRECL 24"
"FILEDEF OUT TERM"
"MOVEFILE IN OUT"
```

This file definition will fail. In this case, the BLKSIZE is not specified. MOVEFILE will calculate the BLKSIZE by adding 4 bytes to the ESTATE LRECL size (19). The characteristics of the file end up being LRECL=24 (specified by the user) and BLKSIZE=23 (calculated from ESTATE information). Since the BLKSIZE must be 4 bytes bigger than the LRECL for variable data, an error will occur when the file is opened.

Note: During MOVEFILE, the RECFM of the CMS file is NOT obtained from the ESTATE. If the user does not specify the RECFM with the FILEDEF command, the default of 'U' Undefined is used. This may produce unpredictable results when working with variable data because LRECL and BLKSIZE verification checks will not be made. RECFM=V should always be specified when working with variable length data.

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 FILE 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 the input file does not exist, MOVEFILE terminates processing.

```
DMSMVE002E  File fn ft fm not found
```

If the input file is empty, the MOVEFILE fails with message:

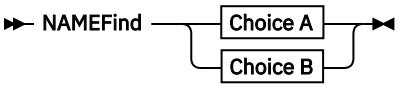
```
DMSMVE2521I  MOVEFILE cannot be performed on empty file fileid
```

MOVEFILE detected the input file was empty before it opened the output file. Processing stops. If it could not detect the input file was empty before opening the output file, MOVEFILE issues this message and both versions of message DMSMVE173E.

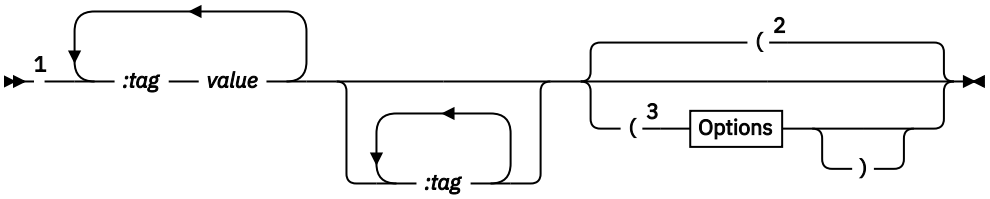
Messages and Return Codes

- DMS002E File[(s)] [*fn* [*ft* [*fm*]]] not found [RC=28]
- DMS003E Invalid option: *option* [RC=24]
- DMS037E Output filemode *mode*[(*vdev*)] is accessed as read/only [RC=36]
- DMS041E Input and output files are the same [RC=40]
- DMS069E Output filemode *mode* is not accessed [RC=36]
- DMS070E Invalid parameter *parameter* [RC=24]
- DMS073E Unable to open file *{ddname|fn}* [RC=28]
- DMS075E Device *devtype* invalid for {input|output} [RC=40]
- DMS086E Invalid DDNAME *ddname* [RC=24]
- DMS089E Open error code *nn* on {*fn*|SYS*aaa*|tap*n*} [RC=36]
- DMS127S Unsupported device for DDNAME [RC=100]
- DMS128S I/O error on input after reading nnn records; input error code on DDNAME [RC=100]
- DMS129S I/O error on output writing record number nnnn; output error code on DDNAME [RC=100]
- DMS130S Blocksize on V format file *ddname* is less than 9 [RC=88]
- DMS173E Empty output file *fileid* not created [RC=40]
- DMS173E Output file may have been erased due to empty condition [RC=40]
- DMS225I PDS member *membername* moved
- DMS226I End of PDS move
- DMS1116E Invalid value *value* for BLKSIZE in *outddname* [RC=32]
- DMS2139I VDEV *vdev* SENSE gives ERA/RAC=*rc*; cartridge may not be valid for I/O
- DMS2521E MOVEFILE cannot be performed on empty file [RC=88]
- DMS3919I Zero length input records not moved to output file.

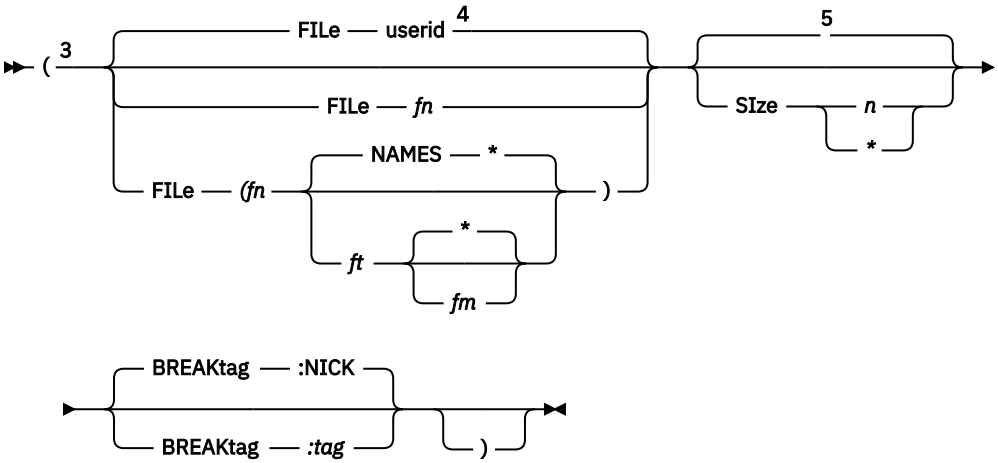
NAMEFIND



Choice A



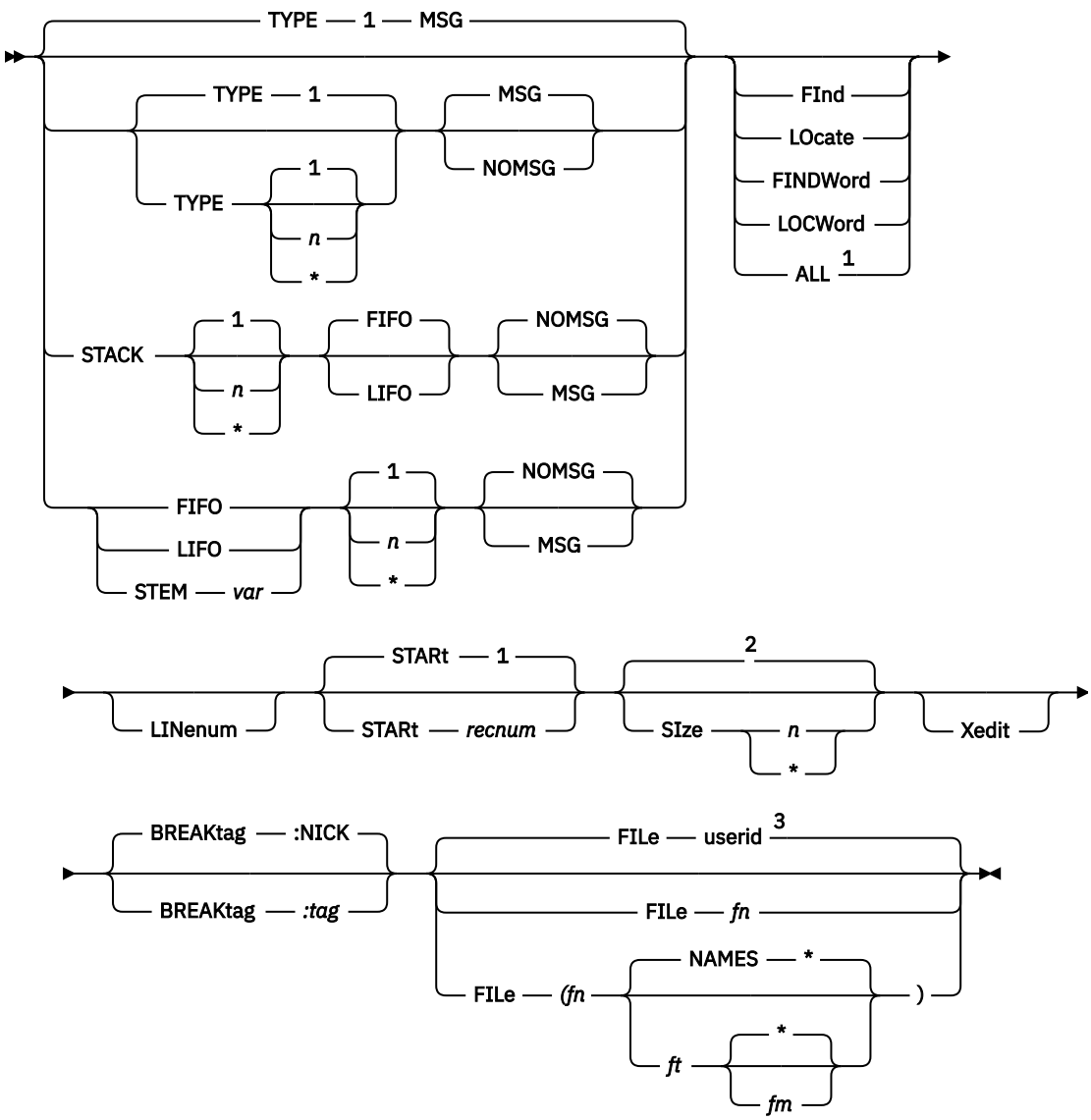
Choice B



Notes:

[1] Search tags and return tags can be entered in any order.
[2] The defaults you receive appear above the line in the Options fragment.
[3] You can enter Options in any order between the parentheses. At least one Option must be specified.
[4] "userid" is your virtual machine user ID.
[5] Default is the last NAMEFIND “SIZE n” specification or “SIZE *”.

Options



Notes:

[1] If you specify the ALL option, search tags should not be entered; and the number of entries (*n*) for the TYPE, STACK, LIFO, FIFO, and STEM options is ignored.

[2] Default is the last NAMEFIND “SIZE n” specification or “SIZE *”.

[3] "userid" is your virtual machine user ID.

Authorization

General User

Purpose

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).

Operands

:tag
is a tag in a names file. You can specify multiple tags in a NAMEFIND command. For more information, see Usage Note “3” on page 520, "Format of a Names File."

Each tag specified with a value is a *search tag*. NAMEFIND searches the names file until either all search tags and values are matched in one or more entries or all entries have been checked.

Each tag specified without a value is a *return tag*, whose value is returned for every entry matching the search criteria. If a return tag does not exist in a matching entry, a null value is returned.

If no return tags are specified, all tag names and values in the matching entries are returned.

Search tags and return tags can be specified in any order.

For more information, see Usage Note “5” on page 523, "How NAMEFIND Searches a Names File and Returns Information."

value
is the value of a search tag.

Options

TYPE *n*
specifies information from the number of entries specified (*n*) that meet the search criteria is displayed at the terminal. Valid values for *n* are any positive integer 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. This option is the default.

STACK *n* FIFO
STACK *n* LIFO
specifies information from the number of entries specified (*n*) that meet the search criteria is placed in the program stack. Valid values for *n* are any positive integer 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. The information is stacked either FIFO (first in first out) or LIFO (last in first out). The default order is FIFO.

FIFO *n*
specifies information from the number of entries specified (*n*) that meet the search criteria is placed in the program stack. Valid values for *n* are any positive integer 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. The options STACK, STACK FIFO, and FIFO are all equivalent.

LIFO *n*
specifies information from the number of entries specified (*n*) that meet the search criteria is placed in the program stack. The information is stacked LIFO (last in first out). Valid values for *n* are any positive integer 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. This option is equivalent to STACK LIFO.

STEM *var n*
specifies information from the number of entries specified (*n*) that meet the search criteria is set into exec variables using the EXECCOMM interface. The valid values for *n* are any positive integer 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 set into exec variables.

The specified variable name (*var*) constructs the exec variables that are set. The variable name is suffixed with a '1' for the first return value, a '2' for the second return value, and so on. The variable name suffixed with a '0' is set to the number of variables set to the return information.

Note: If there is a "." at the end of the variable name, a REXX STEM can be set up as an array.

MSG
specifies all NAMEFIND error messages will be displayed. If this option is omitted and a STACK or STEM option is specified, only error messages relating to the parsing of the command are displayed.

The default is MSG when you specify the TYPE option.

NOMSG
specifies NAMEFIND error messages will not be displayed except for error messages relating to the parsing of the command.

The default is NOMSG when you specify the STACK, FIFO, LIFO, or STEM options.

FInd
specifies tag values in the names file that begin with the search tag values are considered to be a match. An exact match of the search tag values to the entry tag values, in their entirety, is not required.

LOcate
specifies tag values in the names file that completely contain the search tag values are considered to be a match. An exact match of the search tag values to the entry tag values, in their entirety, is not required.

FINDWord
specifies search tag value words will be matched to tag value words in the names file. Each word of the search tag text is matched to each word in the corresponding tag text for an entry. If the tag text word starts with and completely contains the search text word, it is considered to be a match. If all the words in the search text are matched, the entry is considered to be a match. Words are matched in any order, not just the order in which the words in the search tag text are specified.

LOCWord
specifies search tag value words will be matched to tag value words in the names file. Each word of the search tag text is matched to each word in the corresponding tag text for an entry. If the tag text word exactly matches the search text word, it is considered to be a match. If all the words in the search text are matched, the entry is considered to be a match. Words are matched in any order, not just the order in which the words in the search text are specified.

ALL
specifies all entries in the names file, following START *recnum*, will be automatically matched. Only return tags are allowed as operands on the command line with this option. If no return tags are specified, all tags and values for all entries are returned. The (*n*) parameter on the TYPE, STACK, FIFO, LIFO, and STEM options, if specified, is ignored when also using this option.

LINenum
requests the record number of the beginning of each matched entry be returned. The record number is returned before any other information from the matched entry. The record number returned has 9 digits and a value of 000000001 to 999999999.

STARt *recnum*
specifies the search is to begin at the *recnum* record of the names file. *recnum* can be any positive integer. The default is "START 1".

SIze *n*
specifies the maximum amount of storage, in 1024-byte (K) units, to be allocated to buffer the names file. Valid values for *n* are any non-negative integer or *. If zero (0) is specified, the names file is not buffered in storage. If *n* is specified less than the amount of storage needed to fully buffer the names file, the file is partially buffered in storage. If *n* is specified greater than the amount of storage needed to buffer the names file or asterisk (*) is specified, only the amount of storage needed to fully buffer the file is allocated.

If the SIZE option is omitted, the default maximum buffer size for NAMEFIND is assumed.

For more information, see Usage Note “12” on page 531, "Using the SIZE Option."

Xedit
specifies the names file should be read from the XEDIT environment rather than from the NAMEFIND buffer or from disk.

For more information, see Usage Note “16” on page 531, "Using the XEDIT Option."

BREAKtag :*tag*
specifies the breaktag, *:tag*, separates entries in the names file to be processed. If the BREAKTAG option is omitted, the default breaktag is :NICK. For more information on breaktags, see Usage Note “3” on page 520, "Format of a Names File."

FILe *fn*
FILe *(fn ft fm)*
specifies a file whose file name is *fn* and whose file type is *ft* and whose file mode is *fm*. This option allows you to use NAMEFIND to search a names file whose file name, file type, or file mode is something other than your "user ID NAMES *". If the file type is not specified, the default is "NAMES". If the file mode is not specified, the default is "*". If this option is not specified, the "userid NAMES" file is searched.

Usage Notes

1. There are two forms of NAMEFIND invocation:

   Choice A
   Usage Note “5” on page 523, "How NAMEFIND Searches a Names File and Returns Information".

   Choice B
   Usage Note “13” on page 531, "Choice B NAMEFIND Invocation".

2. Overview of Names Files

   A names file is a collection of *entries*, with each entry identified by a *breaktag*. A breaktag plus a series of other tags with their associated values make up an entry. The default breaktag is *:NICK*, this identifies the *nickname*.

   A special names file is one whose file ID is "userid NAMES." The "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 your "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.

   Other CMS commands like GRANT AUTHORITY also allow you to use a nickname to refer to users or groups of users instead of specifying their user IDs.

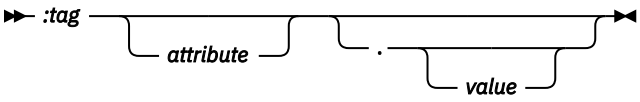
   You can create names files to support functions other than sending messages and data to other users. For example, you could create a names file that consists of system printer definitions. For more information on how CMS supports special names files, see Usage Note “8” on page 530 and Usage Note “9” on page 530.

   You can add, remove, or change entries in your names files either by using the NAMES command (which displays a menu) or by editing the files directly.

3. Format of a Names File

   Data Lines

   Data lines in a names file consist of *tags* whose format is:



   :tag
   A tag name always begins with a colon (:). A tag name imbedded in a line must be preceded by one or more blanks. A tag name can be the first word in a line. A tag name can consist of any characters, but is terminated by a blank, a period (.), or the end of a line. The tag value follows the period delimiter of a tag. One or more blanks preceding the period tag delimiter can indicate the beginning of a tag attribute. Blanks in a tag definition, between the tag name and period delimiter, are ignored by NAMEFIND when searching a names file and returning information. Except for the

breaktag, duplicate tag names can be specified for an entry; they are processed by NAMEFIND for search and return in the order specified.

attribute
An optional tag attribute may follow the tag name and one or more blanks; or if a tag name was specified on a previous line and the period tag delimiter was not specified, any characters preceding the period tag delimiter on following lines are considered part of a tag attribute. A tag attribute can consist of any characters, but is terminated by a period tag delimiter or by the end of the file. A tag attribute can continue across any number of lines. A tag attribute is ignored by NAMEFIND when searching a names file and returning information.

A tag attribute may contain a string enclosed in single quotation marks ('). The string can consist of any characters, including blanks, colons, and periods, but a single quotation mark terminates the string.

value
A tag value is specified after the period tag delimiter. A tag value can consist of any characters, but is terminated by the beginning of the next tag or the end of the file. A tag value need not be on the same line as its tag. A tag value can continue across any number of lines. The value of a tag may be omitted; in that case, the tag is considered to have a null value.

Leading and trailing blanks in a tag value are ignored by NAMEFIND when searching a names file and returning information. Multiple imbedded blanks are considered to be one blank by NAMEFIND when searching a names file; but when the tag value is returned, all imbedded blanks are included. Tag values split across lines are considered by NAMEFIND to have a single blank between the last nonblank character on the preceding line and the first nonblank character on the following line when searching a names file and returning information.

Breaktags

The only tag required is the *breaktag*. This is the primary tag; there is one for each *entry* in a names file. A breaktag identifies the beginning of an entry; a breaktag identifies the end of an entry and begins the next entry. The end of the file is also the end of an entry. Any tags and values that follow a breaktag are part of an entry. There can be any number of tags and values between breaktags.

The default breaktag is *:NICK*. The value of the :NICK tag is called a *nickname*.

```
:NICK.nickname
```

There can be entries with duplicate breaktag values in a names file; they are processed by NAMEFIND for search and return in the order specified in the file.

Note: If a duplicate :NICK entry is submitted from the NAMES panel (which is displayed by the NAMES command), a warning message is displayed.

The breaktag must be the first word on a line. If a breaktag is not the first word on a line, it becomes part of an entry; the processing of an entry with such an "embedded breaktag" may give unexpected results.

Non-data Lines

Other lines in a names file can be:

```
*    An asterisk in column one begins a comment line.
.*   A period in column one and an asterisk in column two
     also begins a comment line.

     Blank lines are ignored.
```

Any nonblank lines at the beginning of a names file that precede the first line with a breaktag are ignored by NAMEFIND.

4. Character Set Considerations for Names Files

Almost any characters can be used in tag names, attributes, and values, depending on the context and the definitions above. But you should try to *not* use the following characters:

```
:  .  '  ;  (  )  @  #  ¢  "  ¦
```

- Colon (:), Period (.), and Single Quotation Mark (')

  These are names file control characters which, along with blanks, are recognized by NAMEFIND as tag, value, and attribute delimiters depending on their placement. For more information, see Usage Note “3” on page 520, "Format of a Names File."

- Parentheses

  You should avoid parentheses in your tag names and values because the NAMEFIND command and other CMS commands regard the left and right parentheses as the beginning and end of command options.

  For instance, in the following example:

  ```
  namefind :phone (123)-4126
  ```

  The NAMEFIND command would regard 123 as a command option, would issue an error message telling you it was not a valid option, and would ignore the rest of the information after the right parenthesis.

- Logical Line editing characters

  If you have CP LINEDIT set ON, you should not use the logical line editing characters defined for your virtual machine. This is a list of the line editing symbols and their default values:

Symbol	Default Value
@	Logical character delete (CHARDEL)
#	Logical line end (LINEND)
¢	Logical line delete (LINEDEL)
"	Logical escape character (ESCAPE)
¦	Logical tab character (TABCHAR)

  For more information regarding the line editing characters, see *z/VM: CP Commands and Utilities Reference*.

- Semicolon (;)

  When you enter values for the address tag (:ADDR) in the NAMES command MAIL or ALTMAIL panel, semicolons may be inserted into the address tag value as new line indicators. These semicolons do not appear in the NAMES panel.

  If you specify the address tag as a search tag, you must include appropriate semicolons in the search tag value for an exact match. The partial match options, such as FIND and LOCATE, can assist in address tag searches.

  Example

  In the sample "userid NAMES" file in Figure 21 on page 532, the address tag value in entry :nick.SNOOZY is:

  ```
  :addr.Dwarf Cottage;Forest
  ```

  On the NAMES panel, the semicolon does not appear. The entry appears as:

```
 ====> NAMES (Mail panel)      File: SNOWHITE NAMES    A0              <====
Fill in the fields and press a PFkey to display and/or change your names file
Nickname: SNOOZY               Notebook:
Userid: SNOOZY
  Node: COTTAGE
                      Name: I. M. Dozing
                     Phone: 777-7777
                   Address: Dwarf Cottage
                          : Forest
                          :
                          :
            List of Names:
                          :
                          :
                          :
                          :
 Tag:               Value:
 Tag:               Value:

1= Help       2= Add      3= Quit       4= Clear       5= Find      6= Change
7= PrevNick   8= NextNick 9= Delete    10= PrevScrn   11= NextScrn 12= Cursor
                                       ======> Screen 1 of 1 <======
====>
                                                          Macro-read 1 File
```

Figure 20. Example of NAMES Screen for MAIL

The command:

```
namefind :addr dwarf cottage;forest :nick
```

displays:

```
SNOOZY
```

But the command:

```
namefind :addr dwarf cottage forest :nick
```

does not find any entry that matches the search criteria.

5. How NAMEFIND Searches a Names File and Returns Information

   Searching a NAMES File

   When you issue a NAMEFIND command, each tag specified with a value is a *search tag*. The combination of all the search tags and their values constitutes the *search criteria*. NAMEFIND searches the file until all the search tags and values are matched to one or more entries or until all entries have been searched. One or more search tags must be specified unless:

   - the ALL option is specified
   - the "Choice B" form of NAMEFIND invocation is specified. For more information, see Usage Note “13” on page 531, "Choice B NAMEFIND Invocation."

   NAMEFIND starts the search with the first line in the file unless the START option is specified. NAMEFIND searches the names file for a single entry that satisfies the search criteria unless the *n* parameter of the TYPE, STACK, FIFO, LIFO, or STEM option is specified.

   Case and multiple blanks are always ignored during the search. The order of the search tags in the command line and the tags in an entry are ignored during the search.

   Generally, the search tags and values must exactly match, in their entirety, the corresponding tags and values in an entry in a names file; however, there are options—FIND, LOCATE, FINDWORD, and LOCWORD—that allow partial matches to tag values in a names file entry. When one of these options is specified, it applies to all search tag values.

   Identical search tags (the values can be different) may be specified, but all the search tags and values must be matched to tags and values in an entry in the names file (in any order) to be a match.

   Returning Information

Each tag specified without a value is a *return tag*, whose value (from a matched entry) is displayed, stacked, or set in an exec variable. Values of return tags are returned in the order specified in the command.

Given a matched entry, for every return tag specified there is a line of output. If a return tag doesn't exist in a matched entry or the value of the tag is null, a null line is returned for the STEM, STACK, LIFO, and FIFO options. If the information is to be returned to the CONSOLE, a blank is returned, not a null. If identical return tags are specified, the values are returned in the order the identical tags occur in the matched entry; if there are more identical return tags than corresponding tags in the matched entry, null lines are returned as output for the excess return tags. Case and multiple blanks are preserved in the values returned.

If no return tags are specified, all the tags and values of a matched entry are returned. There is one line for every tag in the entry and there is a single blank between the returned tag name and the beginning of its value. Case and multiple blanks are preserved in the tag names and values that are returned.

Information is returned from multiple matched entries in the order in which the entries occur in the names file.

Examples

These examples assume the sample "userid NAMES" file in Figure 21 on page 532.

- The command:

```
namefind :nick snow :name :phone
```

  displays:

```
Snow White
ZZZ-ZZZZ
```

  (`:nick snow` is the search tag. `:name` and `:phone` are the return tags.)

- If no return tags are specified, all tags and values in a matching entry are returned. For example, the command:

```
namefind :nick snow
```

  displays:

```
:nick SNOW
:userid SNOWHITE
:node FOREST
:name Snow White
:phone ZZZ-ZZZZ
:addr Forest Primeval
```

- If you specify two or more identical return tags, a value for each of the tags is returned. A null value is returned as a return tag if corresponding return tags do not exist in the matching entry. For example, the command:

```
namefind :nick snow :name :name :name
```

  displays:

```
Snow White


Ready;
```

  (`:nick snow` is the search tag and the three `:name` tags are identical return tags).

  There is only one :NAME tag defined in the matching entry, so the second and third return tags return null lines.

- You can specify ":LIST" as a return tag to display all the names following the "list of names" tag in a "userid NAMES" file. For example, the command:

  ```
  namefind :nick dwarfs :list
  ```

  displays:

  ```
  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 to 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 returned. For example, the command:

  ```
  namefind :userid queen :nick
  ```

  displays:

  ```
  WITCH
  ```

- If multiple matching tag values exist in a single entry of a names file, that entry is only returned once as a matched entry. To find multiple entries that contain a matched tag value, specify the *n* option with an asterisk or a value greater than one. For example, to return the user IDs for all entries with the node of "COTTAGE":

  ```
  namefind :node cottage :userid (type *
  ```

  displays:

  ```
  SNOOZY
  DUMMY
  BOSS
  SNIFFLES
  GROUCHY
  SMILEY
  WISTFUL
  ```

- The value of a search tag must match a file entry tag value in its entirety for the entry to be considered a match unless one of the *partial match* options is specified:
  - FIND

    To return all the user IDs for entries with nodes beginning with "C", the command:

    ```
    namefind :node c :userid (find type *
    ```

    displays:

    ```
    SNOOZY
    DUMMY
    BOSS
    SNIFFLES
    GROUCHY
    SMILEY
    WISTFUL
    QUEEN
    ```

  - LOCATE

    To display the name and full address of the first entry with "arf" in its address tag value, the command:

    ```
    namefind :name :addr :addr arf (locate
    ```

    displays:

```
I. M. Dozing
Dwarf Cottage;Forest
```

– FINDWORD

To display the name and full address of the first entry with words beginning with "cot" and "dw" in its address tag value, the command:

```
namefind :name :addr :addr cot dw (findword
```

displays:

```
I. M. Dozing
Dwarf Cottage;Forest
```

– LOCWORD

To display the nickname of the first entry with the words "dummy" and "grouchy" in its list tag value, the command:

```
namefind :list dummy grouchy :nick (locword
```

displays:

```
DWARFS
```

- The LINENUM and START options can be used to "step through" successive search tag matches in a names file. For example, to return the nicknames of entries with "forest" in their address tag value, the command:

```
namefind :addr forest :nick (locate linenum
```

displays the first match:

```
000000001
SNOW
```

Then the command:

```
namefind :addr forest :nick (locate linenum start 2
```

displays the next match:

```
000000004
SNOOZY
```

and so on.

- The ALL option provides another type of *search criteria*. The ALL option specifies all entries in the names file are automatically matched. Only return tags are allowed with the ALL option; no search tags can be specified. For example, the command:

```
namefind :name (all
```

displays:

```
Snow White
I. M. Dozing
S. A. What
T.O.P. Banana
A. H. Choo
E. B. Scrooge
H. A. Haas
R. U. Shy
Bad Queen
Prince Charming

Ready;
```

The last entry, beginning with :NICK.DWARFS, doesn't have a :NAME tag so a null line is returned.

- The BREAKTAG option can be used to search a names file with a breaktag other than :NICK. The FILE option can be used to search a names file whose file name is other than "userid NAMES". For example, the command:

```
namefind :mytag anyone (breaktag :mytag file myfile
```

searches the file "MYFILE NAMES *" for a match to the following breaktag:

```
:MYTAG.ANYONE
```

6. Tags in a "userid NAMES" File

Many CMS commands use the tags and values in the special names file, "userid NAMES." For example, you can specify a nickname as an operand for the TELL command. The TELL command uses the "userid NAMES" file to resolve the nickname to one or more user IDs and node IDs, and sends the message to the person or list of people.

Other CMS functions make use of the "userid NAMES" file. For example, entries in the CMS APPC/VM private resource registration and authorization NAMES file, "$SERVER$ NAMES", can have an optional :LIST tag which can consist of nicknames resolved using the "userid NAMES" file.

These commands and functions make use of the tags described below. Fields that correspond to these tags appear in the NAMES command MAIL (default) and ALTMAIL panels. 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, and other information are used by the NOTE command to generate note headings.

All 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 one or more of the following tags.

```
:USERID.userid
```

specifies the user ID of the preceding :NICK entry. For example, this tag is used for:

- communicating with a user through the NOTE, SENDFILE, and TELL commands
- determining the file name of the notebook file using the RECEIVE command
- resolving *nickname* operands in CMS commands, such as GRANT AUTHORITY
- authorizing users to access your APPC/VM private resources

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.

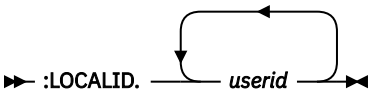
```
:NODE.node
```

specifies the node ID, or VM system ID, corresponding to the user ID tag. If a node tag is not specified, the default is your system ID.

When the :NODE tag value indicates the user is on another system, you should also specify an additional tag and value:

```
:LOCALID.list of userids
```

The following diagram shows you what you need to know about the structure of the tag value:



This tag is used by CMS SFS commands such as GRANT AUTHORITY instead of the :USERID tag when the :LOCALID tag is specified in the entry.

If the :NODE tag value indicates the user is not on the local system and the :LOCALID tag is not specified in the entry, CMS SFS commands such as GRANT AUTHORITY terminate processing with an error message.

If the user is on a system that is part of your TSAF (Transparent Services Access Facility) or CS (Communications Services) collection, the value of the :LOCALID tag is generally the same as the value of the :USERID tag.

If the user is on a system that is not part of your TSAF or CS collection, you will have to ask them for the local user ID that was assigned to them in your TSAF or CS collection. Enter this local user ID as the value of the :LOCALID tag.

Note: 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.

Note: The :LOCALID tag must be entered as a user defined tag and value in the NAMES command MAIL and ALTMAIL panels.

```
: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. For more information on keeping notes, see the NOTE and RECEIVE commands.

```
:NAME.name
```

is the person's real name.

```
:PHONE.phone number
```

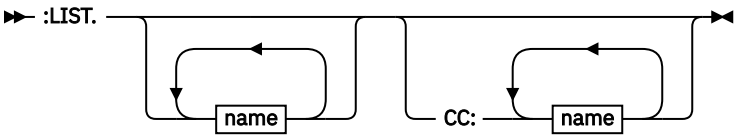
is the person's phone number.

```
:ADDR.address
```

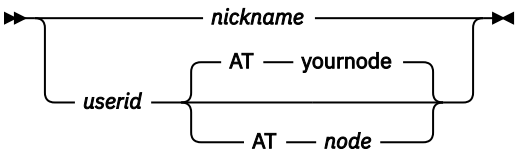
is generally the person's postal address. Semicolons (;) in this tag's value separate the lines of the address. The semicolons do not appear in the "address" fields in the NAMES panel; nor do they appear in the header of a note prepared by the NOTE command.

```
:LIST.list of names
```

is a list of names. The following diagram shows you what you need to know about the structure of the tag value:



name



A name may be a nickname in your "userid NAMES" file. It can be the nickname of one user or the nickname of another list of names. If a name in the list is not a nickname in your "userid NAMES" file, it is assumed to be a user ID on your computer (node). A name can also be specified as *userid AT node*, just as it can in the NOTE, SENDFILE, and TELL commands.

The list can contain `CC:`, followed by one or more names, which is interpreted by the NOTE command as beginning a list of "complementary copy" recipients. Other CMS commands and functions verify `CC:` is followed by one or more names, but otherwise ignore `CC:` in the list.

Attention: Name resolution problems can occur if you have nested :LIST items. For example, :LIST -> :LIST -> LIST :LIST, can result in a duplicate :LIST tag. In this example, PROGS is a :LIST item duplicated in two other :LIST items. Both GROUP1 and GROUP2 could be resolved individually, but ALL, calls out both GROUP1 and GROUP2. When the duplicate PROGS item is used, you will get a names resolution error for GROUP2.

```
:NICK.ALL
     :LIST.GROUP1 GROUP 2
:NICK.GROUP 1
     :LIST.PROGS
:NICK.GROUP 2
     :LIST.PROGS LEADPRG APPLPRG1 APPLPRG2
:NICK.PROGS
     :LIST.SYSPROG TECHPRG
```

7. Maximum Size of Tag Values in a "userid NAMES" File

   CMS commands and functions have limitations on the number of characters in tag values in the "userid NAMES" file. For example, the NAMES command MAIL panel has a nickname field size of 21 characters. If a nickname in the "userid NAMES" file is greater than 21 characters, the NAMES command will only display the first 21 characters. If the entry is modified by NAMES, the nickname is truncated to 21 characters.

   Table 32 on page 529 contains the maximum number of characters of each of the tag values in the "userid NAMES" file for CMS commands and functions.

Table 32. Maximum size (in characters) of tag values in a "userid NAMES" file

Tag Name	NAMES MAIL Panel	NAMES ALTMAIL Panel	NOTE Command	Other Mail Commands	SFS and Private Resources
:NICK	21	8	255	255	255
:USERID	70	8	8	8	8
:NODE	70	8	8	8	8
:NOTEBOOK	8	8	8	8	NU
:NAME	52	52	46	NU	NU
:PHONE	52	52	70	NU	NU
:ADDR	52/line, 4 lines	52/line, 4 lines	132/line	NU	NU
:LIST	255	255	255	255	255
:TAG (User Defined Tag)	12	12	NU	NU	NU
:VALUE	Screen width - 27	Screen width - 27	NU	NU	NU
:LOCALID	UDT	UDT	NU	NU	255, 8 bytes max per user ID

Table 32. Maximum size (in characters) of tag values in a "userid NAMES" file (continued)

Tag Name	NAMES MAIL Panel	NAMES ALTMAIL Panel	NOTE Command	Other Mail Commands	SFS and Private Resources

Definition of table acronyms:

- NU = Not Used
- UDT = User Defined Tag

8. Private Resource Processing

   APPC/VM 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, along with the "userid NAMES" file, validates private resource conversation requests. For more information, see *z/VM: Connectivity*, "$SERVER$ NAMES" file, and APPC/VM private resources.

   The NAMES command, when invoked with the SERVER option, can be used to create and maintain the "$SERVER$ NAMES" file.

   Note: The values of :LIST tags in the "$SERVER$ NAMES" file and the "userid NAMES" file should not exceed 255 bytes.

9. CMS Communications Directory Processing

   APPC/VM 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 default user-level communications directory is called UCOMDIR NAMES. The information collected in the communications directories resolves APPC/VM symbolic destination names when CMS communications directory processing is enabled. Communications directory processing is enabled by the SET COMDIR command. For more information on using CMS communications directories, see *z/VM: CMS Planning and Administration*.

   The NAMES command, when invoked with the COMDIR option, can be used to create and maintain the communications directory files.

10. User defined Names Files

    Users can define names files for any purpose. The NAMES command can be used to create and maintain any NAMES file with a user-defined panel or an editor can be used.

11. NAMEFIND File Buffer

    When NAMEFIND is invoked without the XEDIT option, the names file is read into a buffer in virtual storage. This allows NAMEFIND to process the file in storage instead of reading the file from a minidisk or directory. NAMEFIND preprocesses a buffered names file, optimizing for breaktag searches, to make subsequent calls as fast as possible.

    The maximum amount of storage allocated for the buffer is determined by the SIZE option, if specified. If the SIZE option is omitted, the buffer size is the NAMEFIND default maximum buffer size. For more information, see Usage Note “12” on page 531, "Using the SIZE Option"

    CMS commands that allow one or more nicknames to be specified as operands, such as NOTE, RECEIVE, SENDFILE, TELL, and GRANT AUTHORITY, invoke the NAMEFIND command to search the "userid NAMES" file. Having your "userid NAMES" file kept in a buffer improves the performance of these commands, particularly if the file is large.

    In subsequent calls of NAMEFIND, characteristics of the names file on disk, such as logical record length, 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 reloaded into the buffer. Other factors that can cause NAMEFIND to unload or reload the names file in the buffer are:

    - the value given on the SIZE option
    - the use of the FILE option with a fileid different from the buffered file (if any)

- the use of the BREAKTAG option with a breaktag different from the breaktag of the buffered file (if any)
- dropping the NAMEFIND nucleus extension

When the XEDIT option is specified, the NAMEFIND file buffer is not affected.

12. Using the SIZE Option

    The default maximum buffer file size for NAMEFIND is:

    - SIZE *
      - before NAMEFIND has been invoked for the first time with the SIZE *n* option
      - if the NAMEFIND nucleus extension has been dropped
    - the value *n* in the last SIZE option specified on a NAMEFIND command

    For more information, see Usage Note “11” on page 530, "NAMEFIND File Buffer."

    For the SIZE * option, NAMEFIND allocates as much virtual storage as necessary to fully buffer the names file. You can invoke NAMEFIND specifying the SIZE *n* option, where *n* is decreased to conserve virtual storage. However, if the names file is larger than the amount of storage allocated for the buffer, NAMEFIND only reads as much of the file as will fit into the buffer, and then reads the rest from the minidisk or directory each time it is invoked.

    When the XEDIT option is specified, the SIZE option is ignored because the file is already in storage.

13. "Choice B" NAMEFIND Invocation

    When NAMEFIND is invoked with search tag operands or the ALL option (the "Choice A" form of NAMEFIND invocation), you can also specify any of the other NAMEFIND options. The "Choice B" form of NAMEFIND invocation loads or reloads the NAMEFIND file buffer. If you specify the SIZE option, the NAMEFIND default maximum buffer size is reset.

    In this form of NAMEFIND invocation, no operands are specified and the ALL option is not specified. Only the FILE, SIZE, and BREAKTAG options may be used. If any other options are specified, NAMEFIND considers this to be a "Choice A" (search and return) invocation and NAMEFIND will output an error message and terminate with a return code.

    The options that may be used during a "Choice B" invocation have the same format and functions as described above.

14. Truncation of Output

    In the following cases, if an output line of return information (tag names and values) exceeds 255 characters, the line is truncated:

    - The STACK, FIFO, or LIFO options are specified
    - The STEM option is specified from an EXEC2 exec

    NAMEFIND continues to stack return tags and values. NAMEFIND completes with return code 88.

15. NAMEFIND uses the extended plist for processing command lines. If you are calling NAMEFIND from an assembler language program, you should supply an extended plist. For more information on how an assembler language program can supply an extended plist, see *z/VM: CMS Application Development Guide for Assembler*.

16. Using the XEDIT option

    This option is only valid when NAMEFIND is issued from the XEDIT environment. When the XEDIT option is specified:

    - the SIZE option is ignored
    - one or more search tags and values must be specified
    - a names file currently buffered by NAMEFIND is not affected
    - the file must be in the current XEDIT ring

Note: The interaction of the NAMEFIND command and XEDIT may move the current line of the names file in XEDIT.

17. NAMEFIND runs enabled for console interrupts.
18. If conflicting options are entered (such as, TYPE and STACK), the last one entered (the rightmost) overrides the others.

Examples

Figure 21 on page 532 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
               :LOCALID.BEDROOM
:nick.DUMMY    :userid.DUMMY    :node.COTTAGE
               :name.S. A. What               :phone.777-7777
               :addr.Dwarf Cottage;Forest
               :EYES.crossed
               :LEGS.wobbly
               :VOICE.silent
: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.111-1111
               :LOCALID.frog
:nick.DWARFS
               :list.SNOOZY DUMMY BOSS SMILEY GROUCHY SNIFFLES
                           WISTFUL
```

Figure 21. Sample 'userid NAMES' File

For more examples on using the NAMEFIND command, see *z/VM: CMS Primer*.

Messages and Return Codes

- DMS002E Files [*fn* [*ft* [*fm*]]] not found [RC=28]
- DMS003E Invalid option: *option* [RC=24]
- DMS029E Invalid parameter *parameter* in the option *option* field [RC=24]
- DMS104S Error *nn* reading file *fn ft fm* from {disk or directory|XEDIT} [RC=100]
- DMS156E Record *nnn* not found--the file *fn ft fm* has only *nnn* records [RC=32]
- DMS618E NUCEXT failed [RC=*nn*]
- DMS621E Bad plist: *message* [RC=24]
- DMS622E Insufficient free storage for NAMEFIND [RC=*rc*]
- DMS622W Insufficient free storage for NAMEFIND buffer, processing continues
- DMS633W Returned values were truncated [RC=88]

- DMS634E No value to search for was specified [RC=24]
- DMS635I No entries were found that matched your search criteria [RC=32]
- DMS637E Missing value for the *option* option [RC=24]
- DMS648E Userid value not resolved, check the filenames file[RC=24]
- DMS688E XEDIT option only valid from XEDIT environment [RC=24]
- DMS2539E Search tags can not be specified with 'ALL' option [RC=24]
- DMS2540T *nucext* nucleus extension dropped while running
- DMS2541E *module* reentry not allowed [RC=38]

Note: For some non-zero return codes, such as RC=28 and RC=32, if the STEM *var* option was specified, *var*0 is set to zero.

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in the Shared File System	"File Pool Server Messages" on page 1414
Errors in using a file	"File Error Messages" on page 1418

Return codes:

RC
Meaning

0
Completed without errors

12
Error was found in the names file

13
NUCEXT failed

24
Bad PLIST or FSOPEN error

28
File not found

31
Error caused a rollback of a shared file(s)

32
No entries matched search criteria

38
NAMEFIND reentry not allowed

41
Insufficient free storage

55
APPC/VM communication error

70
SFS sharing conflict

76
SFS authorization error

88
Output truncation

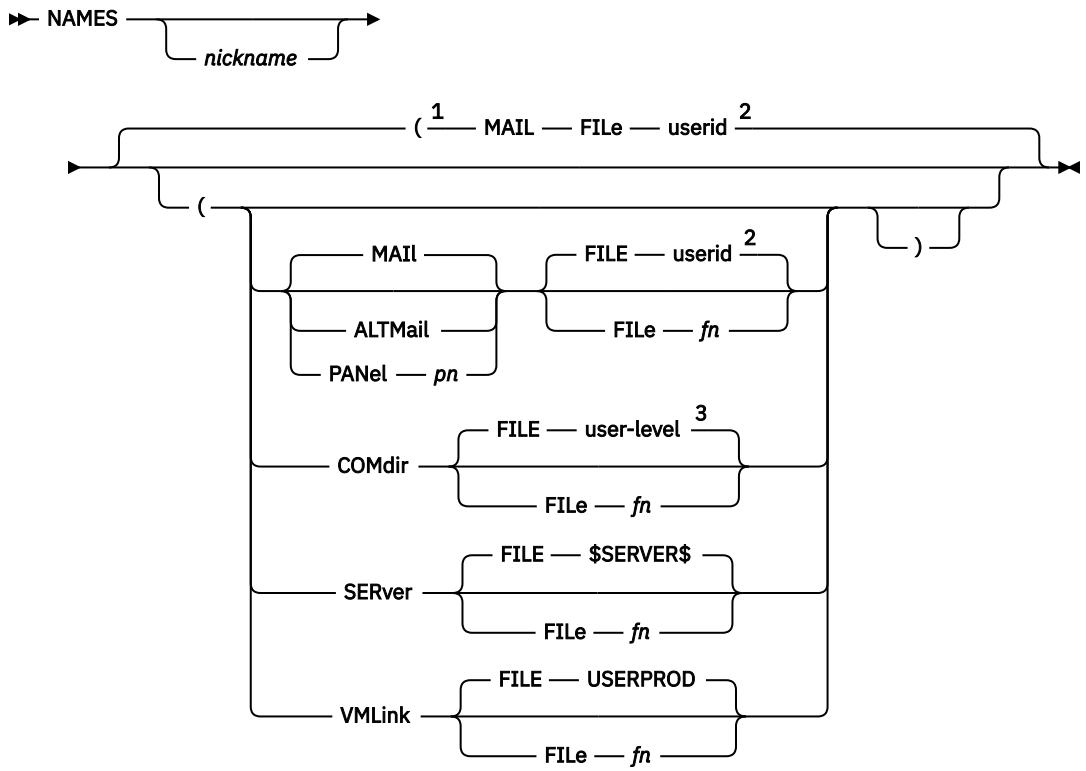
99
Insufficient virtual storage for SFS file pool

100
Error reading file

104
System error

Note: For return codes 28 and 32, if the STEM *var* option was specified, *var*0 is set to zero.

NAMES



Notes:

[1] You can enter Options in any order between the parentheses.
[2] "userid" is your virtual machine user ID.
[3] Default is the file name of the "user-level" communications directory file, if any; otherwise it is "UCOMDIR."

Authorization

General User

Purpose

Use the NAMES command to display a panel from which you can create, change, and remove entries in your names files.

- Electronic Mail names file (such as, "userid NAMES")
- CMS communications directory files (such as, "UCOMDIR NAMES")
- CMS private resources authorization file (such as, "$SERVER$ NAMES")
- VMLINK command names files (such as, "USERPROD NAMES")
- User defined names files

The NAMES panels can only be used on a display terminal.

Operands

nickname
is the name assigned to an entry in a *userid* NAMES file. If you specify a nickname, the NAMES panel is displayed with all the information from that entry (if the entry exists) filled in on the panel. 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 panel is displayed with only the nickname field filled in (with the nickname you specified) truncated to the maximum length the panel will allow. 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 panel is displayed with all fields left blank. You can then fill in the blanks on the panel to create a new entry, or you can scroll through the names file.

Options

MAIl
is the panel tailored for editing your Electronic Mail names file, "*userid* NAMES", where *userid* is your user ID. This is the default NAMES panel. This panel is the same as the ALTMAIL panel except it allows more than 8 characters in the nickname, user ID, and node fields.

For more information, see “Usage Notes” on page 536.

ALTMail
is the alternate panel tailored for editing your Electronic Mail names file, "*userid* NAMES", where *userid* is your User ID. This panel is the same as the MAIL panel except the nickname, userid, and node fields are all 8 characters in length.

PANel *pn*
specifies the file name, *pn*, of a user defined NAMES panel XEDIT macro.

COMdir
is the panel tailored for editing your CMS communications directory files.

SERver
is the panel tailored for editing your CMS APPC/VM private resources authorization names file, "$SERVER$ NAMES".

VMLink
is the panel tailored for editing your VMLINK names file, "USERPROD NAMES."

FILe *fn*
specifies the file name, *fn*, of the names file to be edited. The file type of the names file must be "NAMES" and the file mode is assumed to be "*".

If the FILE option is omitted, the default file name depends on the specified or default panel type.

Usage Notes

1. What Are Names Files?

   A names file is a collection of information about other users with whom you communicate. The file type must be NAMES. The information is arranged using *tag*s, and those tags are arranged into groups called *entries*. Each entry is identified by a special tag value called a *nickname*. An entry in this file is all the information associated with a particular nickname.

   For more information on Names Files, see the NAMEFIND command, “Usage Notes” on page 520.

   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 in the NAMES Command Panels

   The NAMES panels help you create and edit your names files. Each CMS NAMES panel is tailored to the information (tags) in each type of names file supported by CMS. For example, the MAIL and ALTMAIL panels are tailored for your "userid NAMES" file. For more information, see Formats of the CMS NAMES Panels.

   If the information entered in a field spans multiple lines, a blank will be inserted after each line. However, this is not true for the Product Linking Information tag on the VMLINK panel. The information entered for this field will be concatenated.

   All the information you enter on a panel corresponds to an entry in your names file. A panel consists of one or more screens in which you fill in the fields and then press a PF key to create, display, change, or delete the corresponding entry in your names file. For more information on the PF key functions, see Usage Note “10” on page 541, "PF Key Settings on the NAMES Panels."

3. MAIL and ALTMAIL Panels

   The MAIL and ALTMAIL options specify panels tailored for creating and editing your Electronic Mail names file, "userid NAMES." The FILE *fn* option can be used to specify some other names file.

   Note: The MAIL panel is the default panel. The MAIL panel has extended length userid and node fields to allow you to specify a network address in a VM network or another network, such as a TCP/IP network. If you plan to only communicate with other users in a VM network, use the ALTMAIL panel as your default. For more information on how to change this default, see “DEFAULTS” on page 153.

   The following list describes the various fields on these panels and explains the information you type in. For the layouts of these panels, see Figure 30 on page 546 and Figure 32 on page 547.

   Nickname:
   is any name you choose to represent a single user or a list of users. For more information on the screens, see “Examples” on page 542. 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) or refer to this user (in CMS commands such as the GRANT AUTHORITY command).

   You should create an entry for yourself, because the fields that contain your mailing address, phone number, and other information, are used by the NOTE command to generate headings.

   Userid:
   is the user ID of the person whose nickname you specified.

   In the MAIL panel, the userid field is 70 characters long to enable the panel to be used to create and modify nickname entries for non-VM network communications, such as TCP/IP user IDs and node IDs. For VM communications, the userid must be no longer than 8 characters.

   Note: CMS commands such as NOTE and TELL, will terminate processing with an error message and return code for entries in the "userid NAMES" file with a :USERID tag value longer than 8 characters.

   In the ALTMAIL panel, the userid field is 8 characters long.

   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 them through the computer. You might choose to do this if you are using the names file simply to compile an address list.

   Node:
   is the node ID of the person whose nickname you specified. If not specified, the default node ID is your system ID. You can leave this field blank if the nickname represents a list.

In the MAIL panel, the node field is 70 characters long to enable the panel to be used to create and modify nickname entries for non-VM network communications, such as TCP/IP user IDs and node IDs.

For VM communications, the node must be no longer than 8 characters.

Note:

a. CMS commands such as NOTE and TELL, will terminate processing with an error message and return code for entries in the "userid NAMES" file with a :NODE tag value longer than 8 characters.

b. The Node tag cannot be used to specify an IPv6 address that begins with a colon (:). For this case, specify the IPv6 address on the Userid tag.

In the ALTMAIL panel, the node field is 8 characters long.

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.

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 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:
contains 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
- in the form "*userid* AT *node*."

Note:

a. :LIST tag values are limited to 255 characters because these values must be returned through the CMS stack.

b. Because it can be misinterpreted, "AT" should not be defined as a user ID. CMS commands such as NOTE, TELL, and SENDFILE will display an error message if "AT" is interpreted to be a user ID in a command operand or list of names.

c. The list can contain `CC:`, followed by one or more names, which is recognized by the NOTE command as the beginning of a "complimentary copy" list. Other CMS commands verify one or more names follow "CC:" in a list, but otherwise ignore `CC:`.

d. IPV6 addresses that begins with a colon (:) should be entered in the format of userid@node as opposed to 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 Figure 21 on page 532.

Tag:
is an identifier of a tag you can define for your own purposes.

Value:
is the value of your user defined tag identifier above.

4. 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) or CS (Communications Services) 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 or CS collection, the value for the LOCALID tag is the same as the user's user ID. If the user is on a system that is not part of your TSAF or CS collection, you will have to ask them for the local user ID that was assigned to them in your TSAF or CS collection. Enter this local user ID for the value of the LOCALID tag. For an example of how to specify a local user ID for a user on a system in another TSAF or CS collection, see Figure 24 on page 543.

Note: 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. A local user ID cannot be more than 8 characters.

If you are setting up a nickname for a user on your system, no local user ID is necessary.

5. COMDIR Panel

   The COMDIR option specifies the panel tailored for creating and editing your communications directory files. CMS communications directory files are used to define symbolic destination names for APPC/VM routing and access security information. There are two levels of communications directories that can be created or edited using the COMDIR option: system and user. The default system-level communications directory is called "SCOMDIR NAMES"; the default user-level communications directory is called "UCOMDIR NAMES." The information collected in the communications directories is used to resolve your APPC/VM symbolic destination names when CMS communications directory processing is enabled. Communications directory processing is enabled by the SET COMDIR command. Information about your communications directories can be displayed by the QUERY COMDIR command. For more information about CMS communications directories and the tags in these files, see *z/VM: Connectivity*.

   The COMDIR panel has fields corresponding to all of the supported tags in communications directory files. Also, there are **Tag** and **Value** fields for any tags you may add. For more information on the layout of this panel, see Figure 34 on page 548.

   There are some special considerations for the APPC/VM security fields in this panel:

   - When an entry in the CMS communications directory file is read by the COMDIR panel, the entry's APPC/VM security tags are checked. If they contain incorrect or inconsistent information, error or warning messages are displayed. However, the data is displayed as-is to allow it to be corrected.
   - When you add or change an entry, the APPC/VM security fields are checked:
     - If the APPC/VM Security Level is not "NONE", "SAME", or "PGM", an error message is displayed, the entry is not stored, and the entry is redisplayed so you may change it.
     - If the APPC/VM Security Level is not "PGM", and the user ID and the password fields are not null, an error message is displayed, the entry is not stored, and the entry is redisplayed so you may change it. If the userid field is not null, a warning message is displayed.
     - If the APPC/VM Security Level field is "PGM", the userid field is null, and the password is not null, an error message is displayed, the entry is not stored, and the entry is redisplayed so you may change it.

     If the password field is not null for an entry and you have issued the NAMES command with the COMDIR option to view the entry, a warning message is displayed. It is valid if the user ID is not null and password field is null. It is also valid if both the user ID and password fields are null

     The value in the password field is never displayed. If the user types information into the password field, display of the characters is suppressed. A status field immediately below the password field indicates whether the password field contains information.

   If the FILE *fn* option is not specified, the default file name used by the COMDIR panel is the file name of the user-level CMS communications directory, if any. If no user-level CMS communications

directory file is defined or the file type is not "NAMES" or the file mode is not "*", "UCOMDIR NAMES *" is assumed.

6. SERVER Panel

   The SERVER option specifies the panel tailored for creating and editing your APPC/VM private resources authorization file, "$SERVER$ NAMES." The information collected in the $SERVER$ NAMES file is used, along with the "userid NAMES" file, to validate private resource conversation requests. For more information on the APPC/VM private resources and the tags in the "$SERVER$ NAMES" file, see *z/VM: Connectivity*.

   The SERVER panel has fields corresponding to all of the supported tags in the "$SERVER$ NAMES" file. Also, there are **Tag** and **Value** fields for any tags you may add. For more information on the layout of this panel, see Figure 35 on page 549.

   If the FILE *fn* option is not specified, the default file name used by the SERVER panel is "$SERVER$ NAMES".

7. VMLINK Panel

   The VMLINK option specifies the panel tailored for creating and editing your VMLINK names file, "USERPROD NAMES." The information collected in the VMLINK names file provides minidisk linking and SFS directory accessing through a nickname. For more information regarding VMLINK and the tags in the VMLINK names file, see "VMLINK" on page 1143.

   The VMLINK panel has fields corresponding to all of the supported tags in the VMLINK names file. Also, there are **Tag** and **Value** fields for any tags you choose to add.

   If the information entered in the Product Linking Information field spans more than one line, a blank is not inserted at the end of each line to allow for input of SFS directory names up to 154 characters.

   If the FILE *fn* option is not specified, the default file name used by the VMLINK panel is "USERPROD NAMES".

8. Panel *pn*

   You can use the PANEL *pn* option to specify a panel other than the panel types provided with the NAMES command. This gives you the option of creating your own NAMES panel(s) to create and modify the CMS supported names files or your own names files.

   pn is the file name of your NAMES panel XEDIT macro that NAMES will invoke to process your names file. The following list shows which default XEDIT macro is called for each of the NAMES panels provided with the NAMES command.

   Panel type
   XEDIT Macro Profile

   MAIL
   X$NAME$X XEDIT

   ALTMAIL
   X$ONAM$X XEDIT

   COMDIR
   X$CMDR$X XEDIT

   SERVER
   X$SERV$X XEDIT

   VMLINK
   X$NVML$X XEDIT

   When creating your own NAMES panel(s), you should create an XEDIT macro file that uses a format similar to one of these supplied XEDIT profiles.

   If the FILE *fn* option is not specified, the default file name of the names file to edit is "userid NAMES." A panel option specified in a NAMES command overrides the default NAMES panel type specified through the DEFAULTS command.

9. Multiple Screen Considerations

All the CMS NAMES panels are enabled for input in multiple screens for an entry. This allows additional input in some fields (hereafter referred to as "additional input fields"):

- All the NAMES panels allow you to define any number of your own tags and values for your own purposes (**Tag** and **Value** fields).
- Some panels allow you to define list fields that can span multiple screens:
  - MAIL, ALTMAIL, and VMLINK panels: List of Names
  - SERVER panel: Authorization List
  - VMLINK panel: Valid Nodes

At the bottom of every screen, all panels will display:

```
====> Screen n of m <====
```

Where:

n
specifies the number of the currently displayed screen for an entry

m
specifies the total number of screens for the entry.

You can use the PF keys to scroll through the screens. For more information on the PF key functions, see Usage Note “10” on page 541, "PF Key Settings on the NAMES Panels."

The first screen (Screen 1 of *m*) of every NAMES panel type contains all the fields supported by that panel. In the MAIL, ALTMAIL, and VMLINK panels, any subsequent screens have expanded areas for the **List of Names** and **Valid Nodes** fields. You may enter data in any input field in any screen. For the formats of the first and subsequent screens of each NAMES panel type, see Formats of the CMS NAMES Panels.

Data entered in the "additional input fields" is in addition to the data in these fields in previous and subsequent screen(s). To help you keep track of data in these fields, on every screen except the first (Screen 1 of *m*), the first line of each of the "additional input fields" is the same as the last line of the field on the previous screen. Entering data in the first line of these fields will be reflected in the last line of the previous screen when you scroll to it, and vice versa.

Modifications to any other field in any screen will be reflected in all the screens for this panel entry.

You can scroll one screen beyond what is required to display an entry — for example, from Screen *m* of *m* to Screen *m*+1 of *m*+1 — if the last screen has nonblank lines in the "additional input fields." This gives you additional blank fields for input.

For more information on the multiple screens, see “Examples” on page 542.

10. PF Key Settings on the NAMES Panels

    The PF key functions appear on the NAMES panel itself and are summarized:

```
PF  1  Help         Display NAMES command description.
PF  2  Add          Add this entry to the names file.
PF  3  Quit         Exit from the NAMES panel.
PF  4  Clear        Clear all panel fields.
PF  5  Find         Locate all entries that match fields in
                    panel and redefine PF5 to "FindQuit"
                    if more than one matching entry is found.
PF  5  FindQuit     Quit out of the find ring and perform
                    a search if new data has been entered.
PF  6  Change       Change this entry.
PF  7  Previous     Display the previous entry.
PF  8  Next         Display the next entry.
PF  9  Delete       Delete this entry.
PF 10  PrevScrn     Scroll to previous screen for this entry.
PF 11  NextScrn     Scroll to next screen for this entry.
PF 12  Cursor       If cursor is on the panel, move it to
                    the command line; if cursor is on the
                    command line, move it back to its
                    previous location on the panel.
```

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. For more information on the multiple screens, see "Examples" on page 542.

11. Updating a Names File

    You can make changes to the file by using the panel and appropriate PF keys (see above), or by editing the file (for example, `xedit *userid* names`). If you issue NAMES from a line mode terminal, you are placed in edit mode editing the names file.

12. Panel Field Truncation

    If a names file entry is read into a panel with a tag value or a user-defined tag name longer than the corresponding fields on the panel, a warning message is displayed. If you add or change the entry, all associated tag names and tag values are truncated to the length of the panel field before the entry is stored in the names file.

    If you invoke NAMES with a nickname operand, the names file is searched for the first entry that exactly matches the nickname. If a matching entry is found but the nickname is too long to fit in the nickname field, a warning message is displayed. If you modify and save the entry, the entry is replaced with the truncated nickname.

13. List Field Considerations for the MAIL, ALTMAIL, SERVER, and VMLINK panels.

    If an entry is read into the MAIL, ALTMAIL, SERVER, or VMLINK panels with a :LIST or :NODE tag value greater than 255 bytes, an error message is displayed. The entry's :LIST or :NODE tag value is loaded into the "List of Names", "Authorization List", or "Valid Nodes" field as you entered it to allow you to modify it.

    If you attempt to add or change an entry with a "List of Names", "Authorization List", or "Valid Nodes" field greater than 255 bytes, an error message is displayed and the entry is not stored.

14. Use caution when entering certain characters in the panel fields. For example, the colon (:) is a tag delimiter in the names file so you should avoid using it.

    For more information see the Usage Notes for the NAMEFIND command on page "Usage Notes" on page 520.

15. Searching a Names File

    You can search the file by entering the search data in the panel and by using the Find PF key. For more information on the PF key functions, see Usage Note "10" on page 541, "PF Key Settings on the NAMES Panels." All panel fields which have data you entered will be used to find matching nickname entries in the Names file. If no data has been entered and PF5 is executed, the search will not be performed and no error message will be displayed. The screen will be redisplayed as it was before PF5 was entered. If multiple nickname entries are found, these entries for a "find ring". PF7 and PF8 can be used to page through the entries in a find ring. PF5 can be used to either quit the find ring or to quit the find ring and start a new search if new data has been entered.

    Some tag values in the Names file are represented by multiple panel fields. When searching a Names file, the entire tag value is used, not just the lines that have been changed by the user. For example, on the ALTMAIL panel, the tag value associated with the ADDRESS tag is displayed in multiple panel fields. If the ALTMAIL panel is displaying a nickname with an ADDRESS tag value displayed in 3 Address panel fields and the user changes the data in the second Address panel field and uses PF5 to search the Names file, the data from all 3 Address panel fields searches the Names file, not just the data from the field the user changed.

Examples

1. This is an example of the panel you would see if you entered `names` with the nickname operand on your command line (you took the defaults).

```
 ====> NAMES (Mail panel)      File: SNOWHITE NAMES    A0              <====
Fill in the fields and press a PFkey to display and/or change your names file
Nickname: SNOW                  Notebook:
Userid: SNOWHITE
  Node: FOREST
                     Name: Snow White
                    Phone: ZZZ-ZZZZ
                  Address: Forest Primeval
                           :
                           :
                           :
            List of Names:
                           :
                           :
                           :
                           :
 Tag:               Value:
 Tag:               Value:

1= Help       2= Add      3= Quit       4= Clear       5= Find      6= Change
7= PrevNick   8= NextNick 9= Delete    10= PrevScrn   11= NextScrn 12= Cursor
                                       ======> Screen 1 of 1 <======
====>
                                                          Macro-read 1 File
```

Figure 22. Sample NAMES Panel

2. This is an example of the panel you would see if you entered `names` alone on your command line (you took the defaults) and created a nickname for a list of names.

```
 ====> NAMES (Mail panel)      File: SNOWHITE NAMES    A0              <====
Fill in the fields and press a PFkey to display and/or change your names file
Nickname: DWARFS                Notebook:
Userid:
  Node:
                     Name:
                    Phone:
                  Address:
                           :
                           :
                           :
            List of Names:  SNOOZY DUMMY BOSS SMILEY GROUCHY SNIFFLES WISTFUL
                           :
                           :
                           :
                           :
 Tag:               Value:
 Tag:               Value:

1= Help       2= Add      3= Quit       4= Clear       5= Find      6= Change
7= PrevNick   8= NextNick 9= Delete    10= PrevScrn   11= NextScrn 12= Cursor
                                       ======> Screen 1 of 1 <======
====>
                                                          Macro-read 1 File
```

Figure 23. Sample Entry for a List of names

3. The following example is the panel you would see if you entered `names` alone on your command line (you took the defaults) and gave another computer user a nickname, a private notebook, and an additional tag for their entry in your names file.

```
 ====> NAMES (Mail panel)      File: SNOWHITE NAMES    A0              <====
Fill in the fields and press a PFkey to display and/or change your names file
Nickname: GORGEOUS              Notebook: PRIVATE
Userid: PRINCE
  Node: ATLARGE
                     Name: Prince Charming
                    Phone: 111-1111
                  Address:
                           :
                           :
                           :
            List of Names:
                           :
                           :
                           :
                           :
 Tag: LOCALID       Value: frog
 Tag:               Value:

1= Help       2= Add      3= Quit       4= Clear       5= Find      6= Change
7= PrevNick   8= NextNick 9= Delete    10= PrevScrn   11= NextScrn 12= Cursor
                                       ======> Screen 1 of 1 <======
====>
                                                          Macro-read 1 File
```

Figure 24. Sample Entry for a Local User ID

4. The following is an example of what would happen if you wanted to **Find** all the entries that had information that matched all the nonblank fields on the default panel. In this example, PF4 was

pressed to clear the panel and "cottage" was entered in the NODE field as the string to find in the sample names file, "SNOWHITE NAMES." For more information on the sample names file, see “NAMEFIND” on page 516.

```
 ====> NAMES (Mail panel)      File: SNOWHITE NAMES    A0              <====
Fill in the fields and press a PFkey to display and/or change your names file
Nickname:                       Notebook:
Userid:
  Node: cottage
                     Name:
                    Phone:
                  Address:
                          :
                          :
                          :
            List of Names:
                          :
                          :
                          :
                          :
 Tag:               Value:
 Tag:               Value:

1= Help       2= Add      3= Quit       4= Clear       5= Find      6= Change
7= PrevNick   8= NextNick 9= Delete    10= PrevScrn   11= NextScrn 12= Cursor
                                       ======> Screen 1 of 1 <======
====>
                                                          Macro-read 1 File
```

Figure 25. Set up Screen for a Find (PF5)

Pressing PF5 results in the first matching entry in the names file being displayed:

```
 ====> NAMES (Mail panel)      File: SNOWHITE NAMES    A0              <====
Fill in the fields and press a PFkey to display and/or change your names file
Nickname: SNOOZY                Notebook:
Userid: SNOOZY
  Node: COTTAGE
                     Name: I. M. Dozing
                    Phone: 777-7777
                  Address: Dwarf Cottage
                          : Forest
                          :
                          :
            List of Names:
                          :
                          :
                          :
                          :
 Tag: LOCALID       Value: BEDROOM
 Tag:               Value:

1= Help       2= Add      3= Quit       4= Clear       5= FindQuit  6= Change
7= PrevNick   8= NextNick 9= Delete    10= PrevScrn   11= NextScrn 12= Cursor
====> Find 1 of 7 <====              ======> Screen 1 of 1 <======
====>
                                                          Macro-read 1 File
```

Figure 26. Results of Pressing PF5

In this step, we have introduced a Find status area showing "Find 1 of 7." This indicates the NAMES command found 7 nickname entries in the *userid* NAMES file that matched the "cottage" search string entered in the Node input field. These entries form a "Find ring." We are looking at the first entry in the ring which is also the first entry in the "*userid* NAMES" file that had a Node tag value of "cottage."

We can use PF7 and PF8 to page through the entries in the Find ring:

```
 ====> NAMES (Mail panel)      File: SNOWHITE NAMES    A0               <====
Fill in the fields and press a PFkey to display and/or change your names file
Nickname: DUMMY                 Notebook:
Userid: DUMMY
  Node: COTTAGE
                     Name: S. A. What
                    Phone: 777-7777
                  Address: Dwarf Cottage
                         : Forest
                         :
                         :
            List of Names:
                         :
                         :
                         :
                         :
 Tag: EYES          Value: crossed
 Tag: LEGS          Value: wobbly

1= Help       2= Add      3= Quit       4= Clear       5= FindQuit  6= Change
7= PrevNick   8= NextNick 9= Delete    10= PrevScrn   11= NextScrn 12= Cursor
====> Find 2 of 7 <====              ======> Screen 1 of 2 <======
====>
                                                          Macro-read 1 File
```

Figure 27. Results of Pressing PF8

We now have an entry that requires more than one screen (on a 24 line display) to display all of its tags. When we have an entry consisting of more than one screen, we can use PF10 and PF11 to scroll through the screens:

```
 ====> NAMES (Mail panel)      File: SNOWHITE NAMES    A0               <====
Fill in the fields and press a PFkey to display and/or change your names file
Nickname: DUMMY                 Notebook:
Userid: DUMMY
  Node: COTTAGE
            List of Names:
                         :
                         :
                         :
                         :
                         :
                         :
                         :
                         :
 Tag: LEGS          Value: wobbly
 Tag: VOICE         Value: silent
 Tag:               Value:
 Tag:               Value:

1= Help       2= Add      3= Quit       4= Clear       5= FindQuit  6= Change
7= PrevNick   8= NextNick 9= Delete    10= PrevScrn   11= NextScrn 12= Cursor
====> Find 2 of 7 <====              ======> Screen 2 of 2 <======
====>
                                                          Macro-read 1 File
```

Figure 28. Results of Pressing PF11

Notice the first Tag and Value fields in this screen are the same as the last Tag and Value fields on the previous screen. Any modification of this line is reflected in the previous screen when we display it. This helps us keep track of where we are as we scroll through the screens of an entry. The List of Names field has the same feature.

Also, while in "Find mode," PF5 is defined as "FindQuit" if more than one entry matching the field is found; pressing PF5 exits Find mode and returns to normal mode and the Find status message is no longer displayed. The screen then displays the original search field ("COTTAGE" in the Node field) and PF5 is redefined to "Find":

```
 ====> NAMES (Mail panel)      File: SNOWHITE NAMES    A0              <====
Fill in the fields and press a PFkey to display and/or change your names file
Nickname:                       Notebook:
Userid:
  Node: COTTAGE
                      Name:
                     Phone:
                   Address:
                           :
                           :
                           :
            List of Names:
                           :
                           :
                           :
                           :
 Tag:               Value:
 Tag:               Value:

1= Help       2= Add      3= Quit       4= Clear       5= Find      6= Change
7= PrevNick   8= NextNick 9= Delete    10= PrevScrn   11= NextScrn 12= Cursor
                                       ======> Screen 1 of 1 <======
====>
                                                          Macro-read 1 File
```

Figure 29. Results of Pressing FindQuit (PF5)

Notice PF5 is now defined as "Find". If a new search is performed, COTTAGE will be included in the search argument. For example, if the *userid* DUMMY was entered on the screen and PF5 was entered, the search would be performed on the *userid* DUMMY and the node COTTAGE.

Formats of the CMS NAMES Panels

The following sections describe each of the CMS NAMES panels.

MAIL: NAMES Panel For Electronic Mail

```
 ====> NAMES (Mail panel)      File: MYUSERID NAMES    A0              <====
Fill in the fields and press a PFkey to display and/or change your names file
Nickname:                       Notebook:
Userid:
  Node:
                      Name:
                     Phone:
                   Address:
                           :
                           :
                           :
            List of Names:
                           :
                           :
                           :
                           :
 Tag:               Value:
 Tag:               Value:

1= Help       2= Add      3= Quit       4= Clear       5= Find      6= Change
7= PrevNick   8= NextNick 9= Delete    10= PrevScrn   11= NextScrn 12= Cursor
                                       ======> Screen 1 of 1 <======
====>
                                                          Macro-read 1 File
```

Figure 30. Electronic Mail NAMES Panel (MAIL)

Table 33. Fields in the MAIL Panel

Field Name	Maximum Size (in characters)
Nickname	21
Notebook	8
Userid	70
Node	70
Name	52
Phone	52
Address	52/line, 4 lines

Table 33. Fields in the MAIL Panel (continued)

Field Name	Maximum Size (in characters)
List of Names	255
Tag	12
Value	Screen width - 27

Note: LOCALID is specified as a user defined tag.

```
 ====> NAMES (Mail panel)      File: MYUSERID NAMES    A0              <====
Fill in the fields and press a PFkey to display and/or change your names file
Nickname:                      Notebook:
Userid:
  Node:
           List of Names:
                          :
                          :
                          :
                          :
                          :
                          :
                          :
                          :
 Tag:              Value:
 Tag:              Value:
 Tag:              Value:
 Tag:              Value:

1= Help       2= Add      3= Quit       4= Clear       5= Find      6= Change
7= PrevNick   8= NextNick 9= Delete    10= PrevScrn   11= NextScrn 12= Cursor
                                       ======> Screen 2 of 2 <======
====>
                                                        Macro-read 1 File
```

Figure 31. Electronic Mail NAMES Panel (MAIL)

ALTMAIL: Alternate NAMES Panel For Electronic Mail

```
 ====> NAMES (Mail panel)      File: MYUSERID NAMES    A0              <====
Fill in the fields and press a PFkey to display and/or change your names file
Nickname:          Userid:          Node:          Notebook:


                     Name:
                    Phone:
                  Address:
                          :
                          :
                          :
           List of Names:
                          :
                          :
                          :
                          :
 Tag:              Value:
 Tag:              Value:

1= Help       2= Add      3= Quit       4= Clear       5= Find      6= Change
7= PrevNick   8= NextNick 9= Delete    10= PrevScrn   11= NextScrn 12= Cursor
                                       ======> Screen 1 of 1 <======
====>
                                                        Macro-read 1 File
```

Figure 32. Electronic Mail NAMES Panel (ALTMAIL)

Table 34. Fields in the ALTMAIL Panel

Field Name	Maximum Size (in characters)
Nickname	8
Userid	8
Node	8
Notebook	8

Table 34. Fields in the ALTMAIL Panel (continued)

Field Name	Maximum Size (in characters)
Name	52
Phone	52
Address	52/line, 4 lines
List of Names	255
Tag	12
Value	Screen width - 27

Note: LOCALID is specified as a user defined tag.

```
 ====> NAMES (Mail panel)      File: MYUSERID NAMES    A0              <====
Fill in the fields and press a PFkey to display and/or change your names file
Nickname:          Userid:          Node:          Notebook:


            List of Names:
                          :
                          :
                          :
                          :
                          :
                          :
                          :
                          :
 Tag:                Value:
 Tag:                Value:
 Tag:                Value:
 Tag:                Value:

1= Help       2= Add      3= Quit       4= Clear       5= Find      6= Change
7= PrevNick   8= NextNick 9= Delete    10= PrevScrn   11= NextScrn 12= Cursor
                                       ======> Screen 2 of 2 <======
====>
                                                          Macro-read 1 File
```

Figure 33. Electronic Mail NAMES Panel (ALTMAIL)

COMDIR: NAMES Panel For Communications Directories

```
 ====> NAMES (Comdir panel)    File: UCOMDIR  NAMES    A0              <====
Fill in the fields and press a PFkey to display and/or change your names file
Nickname:          Mode Name:
LU Name:
TP Name:

      APPC Security Level:        (NONE,SAME, or PGM)
                  User ID:           (For PGM security only)
                 Password:           (For PGM security only)


 Tag:                Value:
 Tag:                Value:
 Tag:                Value:
 Tag:                Value:
 Tag:                Value:
 Tag:                Value:
 Tag:                Value:

1= Help       2= Add      3= Quit       4= Clear       5= Find      6= Change
7= PrevNick   8= NextNick 9= Delete    10= PrevScrn   11= NextScrn 12= Cursor
                                       ======> Screen 1 of 1 <======
====>
                                                          Macro-read 1 File
```

Figure 34. Communications Directory NAMES Panel

Table 35. Fields in the COMDIR Panel

Field Name	Maximum Size (in characters)
Nickname	8

Table 35. Fields in the COMDIR Panel (continued)

Field Name	Maximum Size (in characters)
Mode Name	8
LU Name	17
TP Name	64
APPC Security Level	4
Userid	8
Password	8
Tag	12
Value	Screen width - 27

SERVER: NAMES Panel For Private Resource Authorizations

```
 ====> NAMES (Server panel)    File: $SERVER$ NAMES    A0              <====
Fill in the fields and press a PFkey to display and/or change your names file
Nickname:          Module to Invoke:


       Authorization List:
                          :
                          :
                          :
                          :

 Tag:               Value:
 Tag:               Value:
 Tag:               Value:
 Tag:               Value:
 Tag:               Value:
 Tag:               Value:
 Tag:               Value:

1= Help       2= Add      3= Quit       4= Clear       5= Find      6= Change
7= PrevNick   8= NextNick 9= Delete    10= PrevScrn   11= NextScrn 12= Cursor
                                       ======> Screen 1 of 1 <======
====>
                                                          Macro-read 1 File
```

Figure 35. CMS private resources NAMES Panel

Table 36. Fields in the SERVER Panel

Field Name	Maximum Size (in characters)
Nickname	8
Module to Invoke	8
Authorization List	255
Tag	12
Value	Screen width - 27

VMLINK: NAMES Panel For VMLINK

```
 ====> NAMES (Vmlink panel)    File: USERPROD NAMES    A0              <====
Fill in the fields and press a PFkey to display and/or change your names file
Nickname:               Description:
Product Linking    :
Information (USERID:
cuu/.DIR dirname)  :
Category:
                  Invoke:
                Preexit:
                   Exit:
             Valid Nodes:
                         :
           List of Names:
                         :
                         :
                         :
 Tag:              Value:
 Tag:              Value:
1= Help       2= Add      3= Quit       4= Clear       5= Find      6= Change
7= PrevNick   8= NextNick 9= Delete    10= PrevScrn   11= NextScrn 12= Cursor
                                       ======> Screen 1 of 2 <======
====>
                                                        Macro-read 1 File
```

Figure 36. VMLINK NAMES Panel

Table 37. Fields in the VMLINK Panel

Field Name	Maximum Size (in characters)
Nickname	8
Description	40
Linking Information	171
Category	21
Invoke	Screen width - 27
Preexit	Screen width - 27
Exit	Screen width - 27
Valid Nodes	255
List of Names	255
Tag	12
Value	Screen width - 27

```
 ====> NAMES (Vmlink panel)    File: USERPROD NAMES    A0              <====
Fill in the fields and press a PFkey to display and/or change your names file
Nickname:               Description:
Product Linking    :
Information (USERID:
cuu/.DIR dirname)  :
             Valid Nodes:
                         :
           List of Names:
                         :
                         :
                         :
                         :
                         :
 Tag:              Value:
 Tag:              Value:
 Tag:              Value:
 Tag:              Value:
1= Help       2= Add      3= Quit       4= Clear       5= Find      6= Change
7= PrevNick   8= NextNick 9= Delete    10= PrevScrn   11= NextScrn 12= Cursor
                                       ======> Screen 2 of 2 <======
====>
                                                        Macro-read 1 File
```

Figure 37. VMLINK NAMES Panel

Responses

```
name has been added to your userid NAMES file.
Entry has been deleted from your userid NAMES file.
Entry has been changed in your userid NAMES file.
No value for Password field for this entry.
Value for Password field is not displayed.
```

The following response is displayed on a line mode terminal:

```
You are now editing your Userid NAMES File.
```

Messages and Return Codes

- DMS006E No read/write filemode accessed [RC=36]
- DMS653E Error executing GLOBALV, rc=*nn* [RC=*nn*]

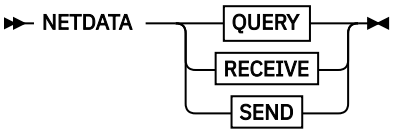
Messages when in the NAMES panel:

- DMS592W Wrapped....
- DMS645W The user tag name *name* is too long to display in the panel
- DMS656E Error searching your NAMES file; rc=*nn* from NAMEFIND command [RC=100]
- DMS658W The value for the *tag* tag is too long to display on the panel
- DMS660E The nickname field must be filled in
- DMS660W Warning: Duplicate nickname entry added. Press the DELETE PFkey if you want this new entry removed. ENTER will redisplay PFKey settings.
- DMS662E You are not on an entry; press PF 5, 7 or 8 to move to an entry
- DMS664E Entry not found
- DMS664E Next entry not found
- DMS664E Previous entry not found
- DMS1007E *list*|*node* is longer than 255 bytes
- DMS1008E *panel* panel not found [RC=28]
- DMS1009E APPC Security Level must be NONE, SAME, or PGM
- DMS1009W APPC Security Level must be NONE, SAME, or PGM
- DMS1013W Entry with null nickname bypassed
- DMS1014E User ID or password only valid with APPC Security Level PGM
- DMS1014W User ID or password only valid with APPC Security Level PGM
- DMS1015E Password not valid without user ID
- DMS1015W Password not valid without user ID

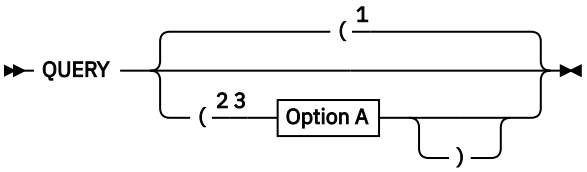
Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

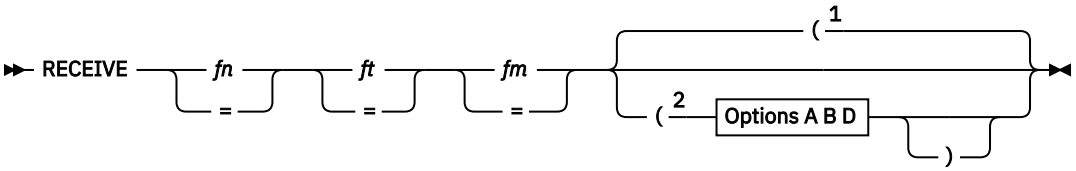
NETDATA



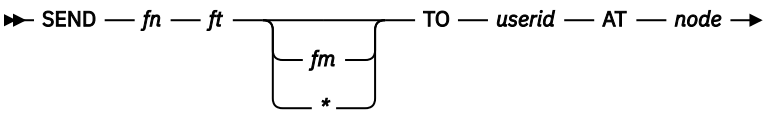
QUERY

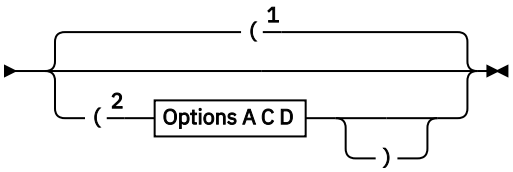


RECEIVE



SEND

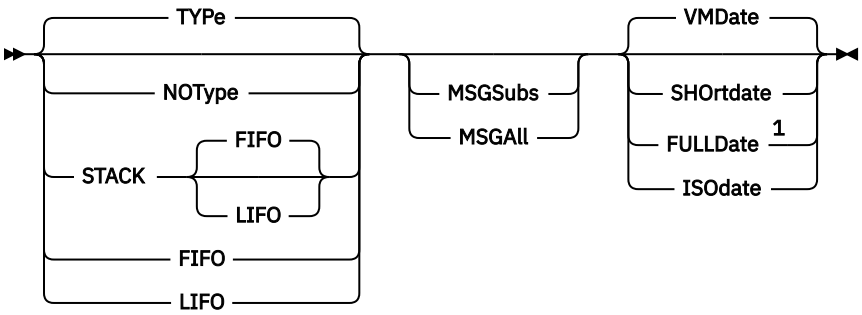




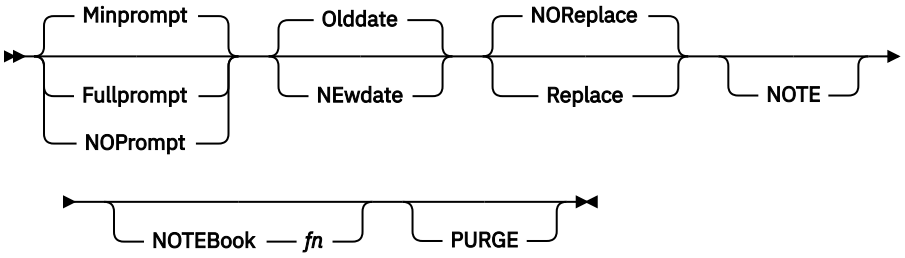
Notes:

[1] The defaults you receive appear above the line in the Options fragments.
[2] You can enter Options in any order between the parentheses.
[3] Option groups may be specified in any order.

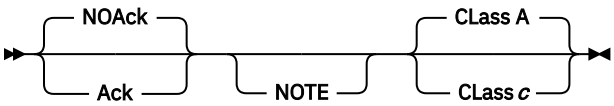
Option A



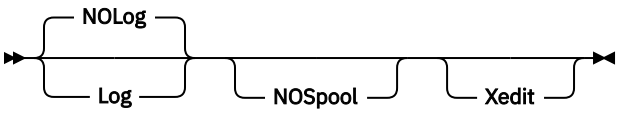
Option B



Option C



Option D



Notes:

[1] Fulldate must be a five-character abbreviation so it does not conflict with fullprompt.

Authorization

General User

Purpose

Use the NETDATA command from within an exec to query, receive, or send files, notes, or acknowledgements in the NETDATA format. Although it is not recommended and is certainly more complex, you may also 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.

Operands

QUERY
requests information about the current reader spool file, if that file is in NETDATA format.

RECEIVE *fn ft fm*
requests 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 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 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 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.

Option A

TYPe
specifies responses will be displayed to the terminal. The TYPE option overrides the FIFO, LIFO, NOTYPE, and STACK options. This is the default.

NOType
specifies responses (excepting prompting messages) will not be displayed to the terminal. The NOTYPE option overrides the FIFO, LIFO, STACK, and TYPE options.

STACK FIFO
STACK LIFO
specifies 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:

1. STACK, STACK FIFO, and FIFO are synonymous.
2. STACK LIFO, and LIFO are synonymous.

LIFO
specifies messages, whether informational or error, are placed in the stack in the inverse order in which they are typed (last in, first out). The LIFO option overrides the FIFO, STACK, and TYPE options and forces the NOTYPE option.

FIFO
specifies messages, whether informational or error, are placed in the stack in the order in which they are typed (first in, first out). 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. For more information on substitution data, see "Usage Notes" on page 556.

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.

VMDate
displays the dates in the format specified by the user's default date format setting. This is the default. For more information on the user's default date format setting, see Usage Note "3" on page 557.

SHOrtdate
displays the dates in *mm/dd/yy* format.

Where:

mm
specifies the month

dd
specifies the day of the month

yy
specifies the 2-digit year

FULLDate
displays the dates in *mm/dd/yyyy* format.

Where:

mm
specifies the month

dd
specifies the day of the month

yyyy
specifies the 4-digit year

ISOdate
displays the dates in *yyyy-mm-dd* format.

Where:

yyyy
specifies the 4-digit year

mm
specifies the month

dd
specifies the day of the month

Option B

Fullprompt
specifies a prompt is to be issued for the incoming file. The FULLPROMPT option overrides the MINPROMPT, NOPROMPT, NOSPOOL, and XEDIT options.

Minprompt
specifies 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 no prompt is to be issued for the incoming file. The NOPROMPT option overrides the FULLPROMPT and MINPROMPT options.

NEwdate
specifies the date and time recorded for the incoming file is that when it was received. The NEWDATE option overrides the OLDDATE option.

Olddate
specifies 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 an existing file is to be replaced by the incoming file. The REPLACE option overrides the NOREPLACE option.

NOReplace
specifies 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 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 the file being received is to be purged, not received.

Option C

Ack
specifies 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 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 the file is to be sent or received as a note.

CLass *c*
specifies the spool class to be used when sending the file. The spool class is a 1-character alphanumeric field whose values can be A through Z, 0 through 9, or equal sign (=). If you specify an equal sign (=), the current PUNCH spool class is used when sending the file. The CLASS option is ignored if the NOSPOOL option was specified. The default is Class A.

Option D

Log
specifies the SEND or RECEIVE operation is to be logged in your *userid* NETLOG file. The LOG option is forced when receiving an acknowledgement. The LOG option overrides the NOLOG option.

NOLog
specifies 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 the status of the punch is not altered during a SEND operation. NOSPOOL also specifies 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. The NOSPOOL option also causes the CLASS option to be ignored during SEND operations.

Xedit
specifies the file being sent should be written from XEDIT storage instead of from a minidisk or directory or the file being received should be read into XEDIT storage instead of to a minidisk or directory. Files read into XEDIT storage will also remain in your reader. The XEDIT option overrides the FULLPROMPT and MINPROMPT options and forces the NOPROMPT option. The new text is the underlined text between the two sets of quoted text which is text that exists today.

When sending a file using the XEDIT option, the contents of the file sent will be from the current line pointer to the end of file, the current date and time at the moment the file is sent will be used as the date the file was last updated. This is the date and time which will be recorded for the file if it is received with the OLDDATE option.

Usage Notes

1. The format of the data transmitted and received by the NETDATA command is described in *z/VM: CMS Macros and Functions Reference*.
2. These messages will be suppressed by the NOTYPE option:
   - NETDATA RECEIVE: responses such as:

```
Note fn ft added to fn NOTEBOOK fm
```

- NETDATA SEND: responses such as:

```
Note fn ft fm sent to userid at node on date time timezone
```

- NETDATA QUERY: would suppress all responses.

3. The default date format used for certain CP and CMS commands can be set on a system-wide basis and also for the individual user. The system-wide default date format is set with the SYSTEM_DATEFORMAT system configuration statement. The user's default date format is set with the DATEFORMAT user directory control statement.

   The system-wide default and the user's default can also be set with the CP SET DATEFORMAT command. If SET DATEFORMAT is not specified for an individual user and there is no DATEFORMAT statement in the user's directory entry, the default format for that user is the system-wide default. The system-wide and user settings can be queried with the CP QUERY DATEFORMAT command. The hierarchy of possible date format settings for the NETDATA command, from highest priority to lowest, is the:

   - NETDATA command option
   - DEFAULTS command setting or default
   - User default
   - System-wide default

4. Tailoring the NETDATA Command Options

   You can use the DEFAULTS command to set up options that override the defaults for the NETDATA RECEIVE and NETDATA SEND commands. However, these customized defaults will be overridden by any options typed in from the command line. For more information, see “DEFAULTS” on page 153.

5. MSGSUBS and MSGALL Options

   The substitution data in a message is the variable information it contains. For example, *fn ft fm*.

   a. The substitution data line generated by the 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.
   b. If NETDATA is issued with the MSGSUBS or MSGALL option and the TYPE, NOTYPE, STACK, FIFO, or LIFO options are not specified, the result is returned to the terminal.
   c. 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.
   d. 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.

6. NETDATA SEND

   Acknowledgements

   Users on different computer systems connected by the RSCS network can request an acknowledgement be returned to them when they send a file, to inform them of its status.

   The sender can specify on the NETDATA SEND command an acknowledgement be returned when a file is received. This 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 options (except the *spoolid* and the PURGE option) are ignored. The acknowledgement makes an entry in your *userid* NETLOG file that confirms the file you sent was received (or discarded). The format of entries in the *userid* NETLOG file is shown in the “Examples” on page 559.

7. Priority

When the NETDATA SEND command sends 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.

8. The default for NETDATA SEND is to send files as CLASS A NOCONT NOHOLD, regardless of the class to which you spool your PUNCH. If you want NETDATA SEND to use the current PUNCH spool class, specify the CLASS = option on the NETDATA SEND command. The CP message generated, containing the *spoolid*, and so forth, is suppressed.
9. NETDATA RECEIVE

   The NETDATA RECEIVE command resets the continuous spooling option and spools your reader NOCONT unless the NOSPOOL option is specified.

   Prompting Messages

   FULLPROMPT, MINPROMPT, or NOPROMPT?:

   - 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.

   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
   - 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 *fn1 ft1 fm1* (or *fn3 ft3 fm3*). For more information on *fn1 ft1 fm1* and *fn3 ft3 fm3*, see “Responses” on page 560.

   2 or Quit
   Ends the command.

   3 or Rename
   Requests prompt message DMS1080R so the incoming file can be received using a different name.

   If you receive prompt message DMS1081R, the valid responses include one of the:

   - Digits specified in the prompt
   - 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 *fn fn fm* and repeats the original prompt message DMS1079R. 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.

Note: You cannot receive a note into a packed notebook. Before using NETDATA RECEIVE to receive the note in the reader, use COPYFILE with the UNPACK option to change the file format from packed to unpacked.

10. Special NETDATA Files from MVS with TSO Extensions (PP)

    The MVS with TSO Extensions licensed program can send an empty file, in which case NETDATA RECEIVE will give you an error message indicating 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.

11. 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 UTC (Coordinated Universal Time).

    For example, suppose a file was last modified on 1 January 1998 at 14:00 in a time zone that is eight hours west of UTC and it is sent to a time zone five hours west of UTC. When the file is received, the time and date are changed to 1 January 1998 at 17:00.

12. Empty Files

    If you try to use NETDATA RECEIVE to receive an empty file into a minidisk, you will get this message:

    ```
    File fileid is empty; minidisk does not support empty files
    ```

    The file is not purged from your reader. You can DISCARD or PURGE the file from your reader or you can try to receive it into an SFS directory. If you receive the file into an SFS directory, the file record length, record format and file ID will be transferred to the new file.

Examples

The *userid* NETLOG file contains an entry for each file sent, received, and discarded. This is an example of a NETLOG file, showing dates in the FULLDATE format.

Note: The date format for the date is the same as the first valid record in your *userid* NETLOG file. If it is a new file, the date format for the date will default to ISODATE (*yyyy-mm-dd*). To change the date formats in your *userid* NETLOG file, see "NETLCNVT" on page 566.

1. Entries in a Sender's NETLOG File:

```
File SMALL    DATA     A1 sent   to MARY     at BROOKLYN on 06/17/2000 18:04:14
Note OHARA    NOTE     A0 sent   to MARY     at BROOKLYN on 06/17/2000 18:15:26
Ackn 06/17/2000 18:05:41  recv   by MARY     at BROOKLYN on 06/17/2000 18:04:14
Ackn 06/17/2000 18:15:41  recv   by MARY     at BROOKLYN on 06/17/2000 18:15:26
Note OHARA    NOTE     A0 sent   to MARY     at BROOKLYN on 06/26/2000 11:05:47
Ackn 06/26/2000 11:14:05  disc   by MARY     at BROOKLYN on 06/26/2000 11:05:47
```

Where:

recv
specifies received. This may be received as a different file ID than when it was sent.

disc
specifies discarded.

Entries in a Recipient's NETLOG File:

```
   File NEW      DATA     A1 recv from OHARA    at CAMBRIDG on 06/17/2000 18:05:26 sent as SMALL DATA
A1
   Note OHARA    NOTE     A0 recv from OHARA    at CAMBRIDG on 06/17/2000 18:15:36
   Note OHARA    NOTE     A0 disc from OHARA    at CAMBRIDG on 06/26/2000 11:12:00
```

2. Examples of date format options of SHORTDATE, FULLDATE, and ISODATE.

   Note: These messages are displayed to the console.

   a. netdata query (fulldate

   ```
   File LRECL 80 recv from MARY at ENDICOTT on 04/17/1997
   08:54:13 sent as TEST FILE A1
   Ready;
   ```

   b. netdata query (isodate

   ```
   File LRECL 80 recv from MARY at ENDICOTT on 1997-04-17
   08:54:13 sent as TEST FILE A1
   Ready;
   ```

   c. netdata query (shortdate

   ```
   File LRECL 80 recv from MARY at ENDICOTT on 04/17/97
   08:54:13 sent as TEST FILE A1
   Ready;
   ```

   d. netdata query (msgall fulldate

   ```
   File LRECL 80 recv from MARY at ENDICOTT on 04/17/1997
   08:55:03 sent as TEST FILE A1
   871002 TEST  FILE     A1 *        *        *  MARY ENDICOTT
       04/17/1997     08:55:03 *        *
       80 *        *     *  TEST FILE A1
   Ready;
   ```

   e. netdata query (msgsubs fulldate

   ```
   871002 TEST  FILE     A1 *        *        *  MARY ENDICOTT
       04/17/1997 08:55:25 *        *
       80 *        *     *  TEST FILE A1
   Ready;
   ```

   f. netdata send test file a to mary at endicott (ack fulldate

   ```
   File TEST FILE A1 sent to MARY at ENDICOTT
   on 04/17/1997 08:55:58
   Ready;
   ```

   g. netdata receive a a a (purge fulldate

   ```
   Ackn 04/17/1997 09:01:19 has been discarded
   Ready;
   ```

Responses

1. If you specify the FULLPROMPT or MINPROMPT option, one of these prompts is displayed:

   ```
   DMS1079R  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 DMS1079R, you will receive the prompting message

```
DMS1080R  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 DMS1080R with a file ID that names an existing file, you will receive the prompting message

```
DMS1081R  Replace fn ft fm?
            Reply 0 (NO), 1 (YES), or 2 (QUIT)
```

4. Successfully receiving a spool file will cause one of these 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 *fn ft* has been discarded
   - Note *fn ft* has been discarded
   - Note *fn ft* 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 this response format when returning the substitution data. This response has a fixed format and fixed length. The length, including blank delimiters and the ending blank, is 184 characters when a date is in the SHORTDATE format. The length, including blank delimiters and the ending blank, is 188 characters when a date is in the FULLDATE or ISODATE format.

   Note:

   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 4-digit message number concatenated with a 2-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 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 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 these commands:

- 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 these commands:

- 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 date format is SHORTDATE or ten when the date format is FULLDATE or ISODATE when you issue these commands:

- 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 you issue these commands:

- 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 the date format is SHORTDATE or ten when the date format is FULLDATE or ISODATE when you issue these commands:

- 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 issue these commands:

- 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 these commands:

- 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*.

  Example

  If TEST FILE A1 exists and you issue this 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:

- 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.xxxxx*___ up to 44 characters in length).
- NETDATA SEND is not set for this field; therefore, it specifies the initialized value of *.

7. NETDATA QUERY Output

These examples use the same format variables as Usage Note “6” on page 561.

Note: All the dates are displayed in the SHORTDATE date format because the user's virtual machine setting is set to SHORTDATE.

- NETDATA QUERY output for an acknowledgment file has this format:

```
Ackn logdate2 logtime2 recv|disc by userid at nodeid on logdate1 logtime1
```

For example:

```
Ackn 07/10/98 07:48:20 disc by SMITHRM at GDLVMA on 07/10/92 07:45:33
```

- NETDATA QUERY output for a file has this format:

```
File LRECL lrecl recv from userid at nodeid on logdate1
   logtime1 sent as sentname
```

For example:

```
File LRECL 63 recv from SMITHRM at GDLVMA on 07/10/98 07:45:33
sent as SAMPLE EXEC B1
```

- NETDATA QUERY output for a note has this format:

```
Note LRECL lrecl recv from userid at nodeid on logdate1 logtime1
```

For example:

```
Note LRECL 132 recv from SMITHRM at GDLVMA on 07/10/98 07:45:33
```

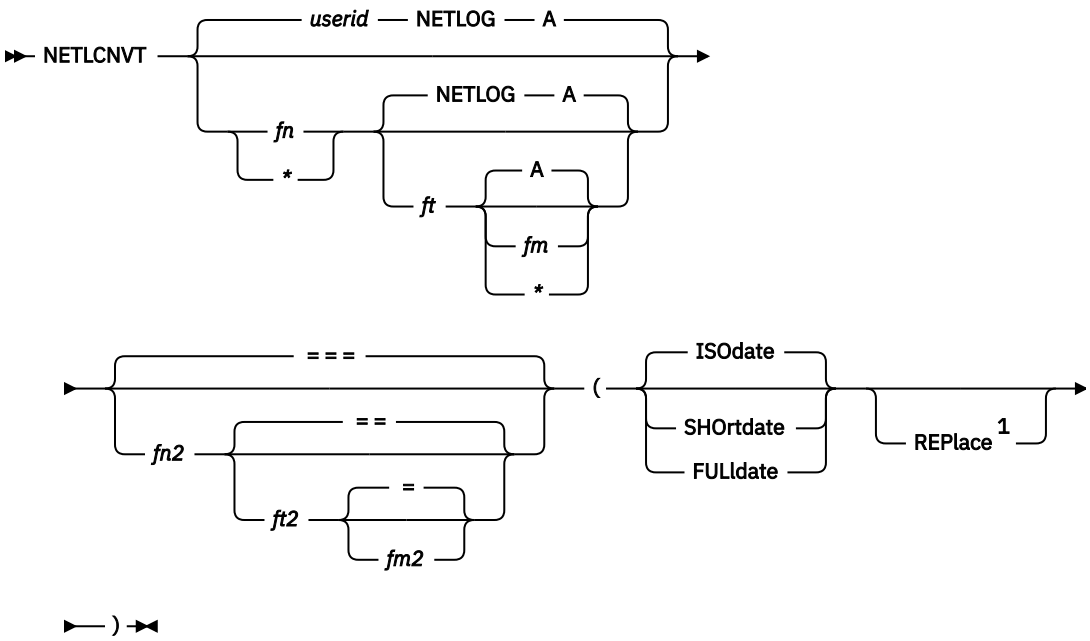
Messages and Return Codes

- DMS006E No read/write filemode accessed for *fn ft* [RC=36]
- DMS024E File *fn ft fm* already exists; specify REPLACE option [RC=28]
- DMS037E Filemode *mode* accessed as read/only [RC=12|36]
- DMS062E Invalid character *char* in fileid *fn ft* [RC=20]
- DMS069E Filemode *mode* not accessed [RC=36]
- DMS078E Invalid card in reader deck [RC=32]
- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS205W Reader empty, reader not ready or empty reader file [RC=100]
- DMS257T Internal system error at address *hex* (offset *hex*)
- DMS636E Unsupported type of NETDATA file [RC=88]
- DMS636W File *fileid* is empty; minidisk|filepool *filepoolid* does not support empty files [RC=74|88]
- DMS638E *entry type* is too wide to append to *fn ft* [RC=32]
- DMS639E Error in *name* routine; return code was *nnnnnnn* [RC=100]
- DMS688E XEDIT option only valid from XEDIT environment [RC=24]
- DMS743E File *fn ft fm* is in an invalid format [RC=40]
- DMS1123E Unknown response *text* ignored
- DMS1124W Spool file *spoolid* has been left in your reader because one or more files were not received [RC=1]
- DMS1138E Filesharing conflict involving file *fn ft fm* [RC=31 | 55 | 99 | 100]
- DMS1184E File *fn ft fm* not found or you are not authorized for it [RC=28]
- DMS1262S Error *nnn* closing | opening file *fn ft fm* [RC=25 | 28 | 30 | 31 | 37 | 40 | 55 | 70 | 80 | 81 | 82 | 83 | 84 | 99]
- DMS1285S Default option *option* is invalid [RC=24]
- DMS514E Return code *nn* from NETDATA [RC=24]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411
Errors in the Shared File System	“File Pool Server Messages” on page 1414

NETLCNVT



Notes:

[1] The default is REPLACE, if the fileid2 is = = = or if fileid2 was not specified.

Authorization

General User

Purpose

Use the NETLCNVT command to convert existing user ID NETLOGs that contain 2-digit years to 4-digit years, or 4-digit years to 2-digit years.

The entries in the *userid* NETLOG file may contain mixed date formats because of non-Year 2000 ready tools or applications writing to them. These are usually in the SHORTDATE date format of *mm/dd/yy*. If this should occur, you can issue the NETLCNVT command to convert the *userid* NETLOG file to contain all the same date formats. Contact your system administrator to update the tools used for the NETLOG file.

Options

***userid* NETLOG A**
: identifies the user ID of the person who owns the NETLOG file. The default is the user ID where the NETLCNVT command is issued.

fn
: specifies the file name of the NETLOG file to be converted. The file name defaults to the user ID issuing the command. If an asterisk (*) is coded in this field, all file names are used.

ft
: specifies the file type of the NETLOG file to be converted. The file type defaults to NETLOG.

fm
: specifies the file mode of the NETLOG file to be converted. The file mode defaults to A. If an asterisk is coded, all accessed disks and directories are searched.

fn2
specifies the file name of the output NETLOG file that will contain the results of the NETLCNVT command.

fn2 ft2 fm2 defaults to:

```
= = =
```

respectively. The equal sign (=) may be coded for any of the *fn2, ft2, or fm2* parameters, indicating it is the same as the corresponding component in the first file ID specified.

ft2
specifies the file type of the output NETLOG file that will contain the results of the NETLCNVT command.

fm2
specifies the file mode of the output NETLOG file that will contain the results of the NETLCNVT command.

ISOdate
specifies the date(s) should be converted to *yyyy-mm-dd* format. This is the default.

Where:

yyyy
specifies the 4-digit year

mm
specifies the month

dd
specifies the day of the month

SHOrtdate
specifies the date(s) should be converted to *mm/dd/yy* format.

Where:

mm
specifies the month

dd
specifies the day of the month

yy
specifies the 2-digit year

FULldate
specifies the date should be converted to *mm/dd/yyyy* format.

Where:

mm
specifies the month

dd
specifies the day of the month

yyyy
specifies the 4-digit year

REPlace
specifies the output file ID to replace an existing file with the same file identifier. The default option is REPLACE when only one file ID is entered or when the output file ID is specified as:

```
= = =
```

Usage Notes

1. The input file and optional output file can have any file name, file type and file mode. If an output file:
   - Is not specified, the input file will be replaced.
   - Already exits and the output file ID is something other than:

     ```
     = = =
     ```

     the user must specify REPLACE.
2. The NETLCNVT command only converts valid NETLOG records. The input file can contain mixed date formats, but they must be one of the following valid date formats:
   - *mm/dd/yy*
   - *mm/dd/yyyy*
   - *yyyy-mm-dd*

   All records that are not valid NETLOG records are copied as is to the output file. The blank lines are not copied to the output file, they are discarded.
3. After the NETLCNVT command is issued, all the dates in the file will be converted to the date format specified.
4. After the conversion of the user ID NETLOG file, all the following entries written by CMS commands will be in the same date format as the first valid NETLOG record.
5. If a user ID NETLOG file is new, all new date entries will be in the ISODATE *yyyy-mm-dd* format. In earlier releases, a new NETLOG file was written in the shortdate *mm/dd/yy* format.
6. This command does not respect the virtual machine date setting.
7. The CMS DEFAULTS command has no effect on the date formats in the user ID NETLOG file.
8. If a user has mixed date formats in the user ID NETLOG file, this command can be run to convert the NETLOG, so all the records have the same date format as specified by the user.
9. A sliding window is used when converting a NETLOG file containing 2-digit year dates as input to a file containing dates with 4-digit years as output.

   In this technique, a given 2-digit year is assumed to reside in the 100-year span [*cy-50*,*cy+49*], where integer *cy* is the current year—that is, the year at the moment of the call.

   For example, if a 2-digit year of 05 is being converted, and the current year (*cy*) is 1997, the window range is [1947,2046]. In this case, the calculation changes the 2-digit year of 05 to the 4-digit year 2005.
10. Existing NETLOGs in SFS directories that are replaced retain the same authorities, aliases, and extended file attributes that already existed.

    New output NETLOG files in SFS directories have none of the original file's authorities, aliases, and will have the current system defaults for the extended file attributes.

    If you create a new NETLOG in another user's directory, the owner of the directory becomes the owner of the file. You, as the creator of the file, automatically have write authority to it.

Responses

The user ID NETLOG file is written to by many CMS commands. SENDFILE, RECEIVE, NETDATA SEND, NETDATA RECEIVE, NOTE, and DISCARD all write entries to the NETLOG file.

Example 1

To convert OHARA's NETLOG file entries to ISODATE (*yyyy-mm-dd*), the user OHARA would enter:

```
netlcnvt
```

This is an example of OHARA's NETLOG file that was converted to ISODATE format:

```
File SMALL    DATA     A1 sent to MAUREEN at BROOKLYN on 2000-06-17 18:04:14
Note OHARA    NOTE     A0 sent to MAUREEN at BROOKLYN on 2000-06-17 18:15:26
Ackn 2000-06-17 18:05:41  recv by MAUREEN at BROOKLYN on 2000-06-17 18:04:14
Ackn 2000-06-17 18:15:41  recv by MAUREEN at BROOKLYN on 2000-06-17 18:15:26
Note OHARA    NOTE     A0 sent to MAUREEN at BROOKLYN on 2000-06-26 11:05:47
Ackn 2000-06-26 11:14:05  disc by MAUREEN at BROOKLYN on 2000-06-26 11:05:47
```

Example 2

If a system administrator wanted to convert all the NETLOG files for that system to ISODATE format, the administrator would link read/write to each minidisk, and access each minidisk or SFS directory (that is used as an a-disk). The administrator can then issue the NETLCNVT command with wildcarding in the file name and file mode operands to convert the NETLOGs on all the accessed minidisks or directories. For example,

```
cp link user1 191 291 w user1pw
Ready;
cp link user2 191 391 w user2pw
Ready;
access 291 b
Ready;
access 391 c
Ready;
access fpool1:user3.myadisk d (forcerw
Ready;
listfile * netlog *
ADMIN1   NETLOG   A0
USER1    NETLOG   B0
USER2    NETLOG   C0
USER3    NETLOG   D0
Ready;
netlcnvt * netlog *
Ready;
```

This will convert the administrator's, USER1, USER2, and USER3's NETLOGs to the ISODATE format. The default is REPLACE because the fileid2 was not specified; the existing NETLOG file will be replaced with the converted NETLOG files.

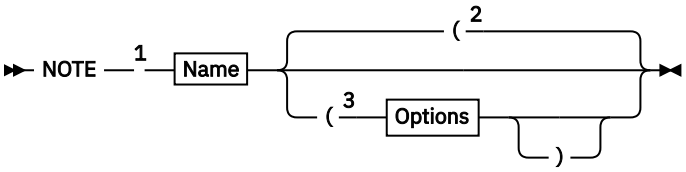
Messages and Return Codes

- DMS002E File *fn ft fm* not found [RC=28]
- DMS024E File *fn ft fm* already exists; specify REPLACE option [RC=28]
- DMS065E *option* option specified twice [RC=24]
- DMS066E *option1* and *option2* are conflicting options [RC=24]
- DMS173E Empty output file *fn ft fm* not created [RC=40]
- DMS208E File *fn ft fm* is not variable record format [RC=40]
- DMS2763E File mode not accessed or not accessed read/write [RC=36]

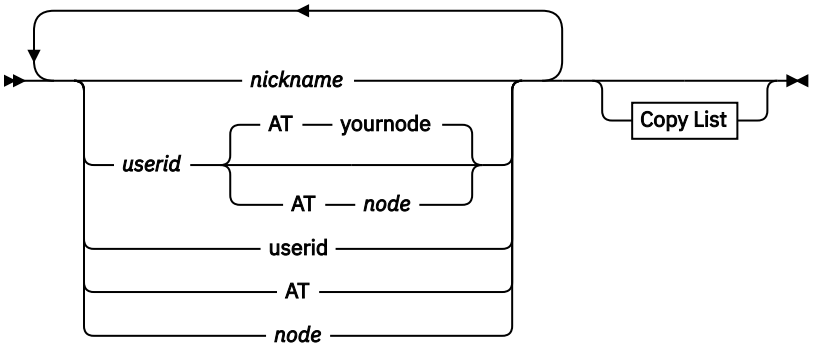
Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

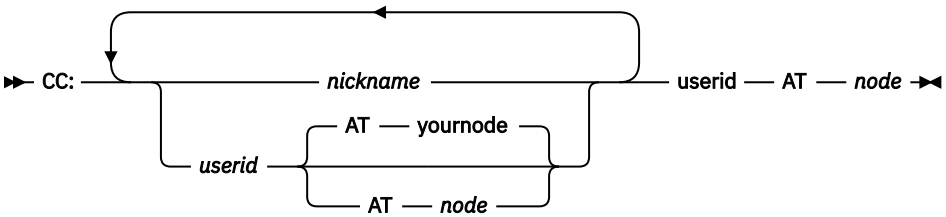
NOTE



Name



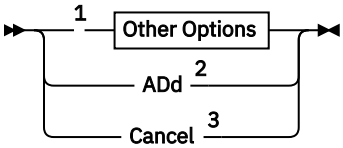
Copy List



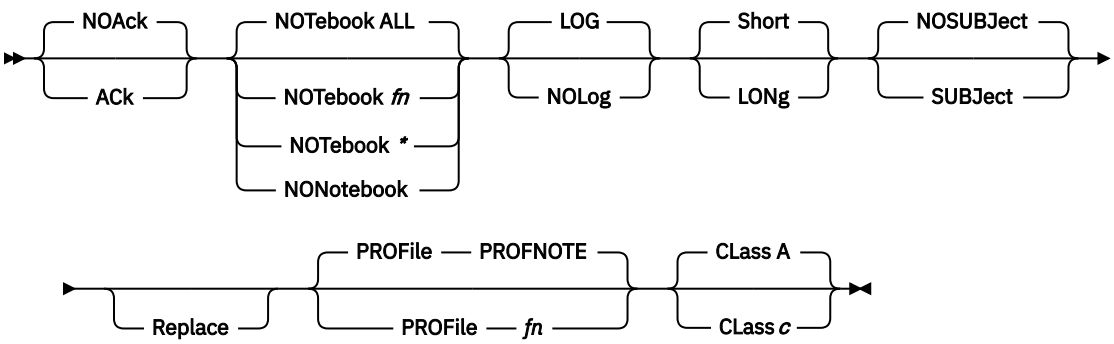
Notes:

[1] If you want to return to an existing note, do not specify any operands or options.
[2] The defaults you receive appear above the line in the Options fragment.
[3] You can enter Options in any order between the parentheses.

Options



Other Options



Notes:

[1] These options are ignored when entered from the NOTE panel command line unless REPLACE is specified. They are also ignored (except for PROFILE *fn*) when returning to an existing note unless REPLACE is specified.

[2] When specifying the ADD option, it must be specified on the NOTE panel command line and no other options may be specified with it. Also, you must specify a nickname, user ID, a CC: list, or a combination of these.

[3] The CANCEL option may only be specified on the NOTE panel command line. Also, the CANCEL option overrides all other options, so should be entered alone with the NOTE command.

Authorization

General User

Purpose

Use the NOTE command to prepare a *note* for one or more computer users on your system or on other computers. A *note* is a short communication, the kind usually done by letter. Some of the features of the NOTE command are:

- XEDIT controls the environment in which a note is prepared, so the full power of XEDIT 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 "NAMES" on page 535.
- 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. For more information, see "Examples" on page 577.
- PF keys are assigned to frequently used functions like sending the note, tabbing, calling for HELP, and so forth.

Operands

name
is one or more *names* of computer users to whom the note is to be sent. If the same recipient is specified more than once, that recipient will be included only once in the list. 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* identifies your user ID. This nickname may represent a single person (on your system or on another system), or a list of several people.

- a user ID of a user on your system. If a name cannot be found in the *userid* NAMES file, it is assumed to be a user ID of a user on your system.
- *user ID* AT *node*, which identifies a user (*userid*) on your system or another system (*node*).

  A name cannot be AT or CC:.

  Note: A node identifies any RSCS name or TCP/IP host name or IP address. If a node contains a period, colon, or dot, it will be treated as a TCP/IP address. A TCP/IP host name that ends with a period will have the local domain name appended to it in order to construct a fully-qualified host name.

  Non-TCP/IP destinations are sent through CMS or RSCS as necessary, even on the same SENDFILE command. You may freely intermix the :pv.userid:epv. and :pv.nickname:epv. forms to specify recipients.

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 continues a note that was started previously. For more information on saving and continuing notes, see Usage Note “5” on page 574, "Continuing Notes".

Options

ACk
requests an acknowledgment be sent to you when the addressee receives your note. This parameter will be ignored for files destined for domain names. For more information on acknowledgments, see “RECEIVE” on page 806.

NOAck
requests 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 Usage Note “6” on page 574, "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.

For more information on the *userid* NAMES file and the NOTEBOOK file, see “NAMEFIND” on page 516 and “NAMES” on page 535.

NOTebook *
specifies the note is to be saved in a file named "*name* NOTEBOOK". To determine the value of *name*, your *userid* NAMES file is searched first for an entry that matches each recipient's user ID and node ID. If a matching entry:

1. Is found and it contains a notebook file name, that file name is used.
2. Is found but it does not contain a notebook file name, the recipient's nickname is used.

3. Was not found, the recipient's user ID is used.

When there is more than one recipient, the full text of the note is saved in the NOTEBOOK file of the first addressee. 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 described above.

NONotebook
specifies a copy of the outgoing note is not to be saved.

LOG
specifies 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.

The date format for the date is the same as the first valid record in your *userid* NETLOG file. If it is a new file, the date format for the date will default to ISODATE (*yyyy-mm-dd*). To change the date formats in your *userid* NETLOG file, see "NETLCNVT" on page 566.

NOLog
specifies this note is not to be logged.

LONg
causes the long form of the note header to be used. For more information, see "Examples" on page 577.

Short
causes the short form of the note header to be used. For more information, see "Examples" on page 577. This is the default.

NOSUBJect
specifies this note will not have a subject line. This is the default.

SUBJect
specifies this note will have a subject line added to the note following the CC: line or the TO: line if no CC: line is present.

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 Usage Note "9" on page 574, "Default PF Key Settings".

CLass *c*
specifies the spool class to be used when sending the note. The spool class is a 1-character alphanumeric field whose values can be A through Z, 0 through 9, or an equal sign (=). If you specify an equal sign (=), the current PUNCH spool class is used when sending the file.

Usage Notes

1. For more information on how to customize this command see Appendix A, "Customizing Profiles for CMS Productivity Aids," on page 1419.
2. The note to be sent begins with the Date: line. With the exception of the USEROPTIONS: line, see USEROPTIONS:, Usage Note "12" on page 575, when you press PF5 to send the note and you have typed any nonblank character between the OPTIONS line and the "Date:" line, you will get an error message, your note will not be sent, and the NOTE panel will be redisplayed with your note.
3. Composing the Note

   When you enter the NOTE command, the note screen appears with the headings. For more information, see "Examples" on page 577. 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.)

4. 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.

5. 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*.

6. 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.

   Be aware that if you specify duplicate recipient names on a NOTE command line, only one is added to the list. However, any existing To: and cc: lists in the note are not checked for duplicates if you subsequently add names with ADD.

   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 choose deleted from the header lines. If you have used the LONG option so the name takes up an entire line, do not blank out the name and create a blank line, or your note will only be sent to those names above the blank line. The names below the blank line will be assumed to be part of the body of the note.

7. Naming Conventions for Userid and Node

   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.

8. Conflicting Options

   If conflicting options are entered (such as ACK and NOACK) the last one entered (the rightmost) overrides the others.

9. 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:

PF1	Help	Display NOTE command description.
PF2	Add line	Add a blank line after the line containing the cursor.
PF3	Quit	Quit this note. The following message may be displayed: FILE HAS BEEN CHANGED. USE QQUIT TO QUIT ANYWAY.
PF4	Tab	Tab the cursor.
PF5	Send	Issue SENDFILE with the NOTE option.
PF6	?	Display the last command issued.
PF7	Backward	Scroll back one screen.
PF8	Forward	Scroll forward one screen.
PF9	=	Repeat the last command issued.
PF10	Rgtleft	Shift the view to the right; press again to shift back to original display.
PF11	Spltjoin	Split a line or join two lines, at the cursor.
PF12	Power® input or Input	If DBCS strings are not supported, enter power typing mode (XEDIT subcommand POWERINP). If DBCS strings are supported, enter input mode (XEDIT subcommand INPUT).

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.

10. For more information on the format of the file created by NOTE and sent by the SENDFILE, see “SENDFILE” on page 888, "Format of the File Sent by SENDFILE".
11. You cannot start a new note while in NOTE.
12. 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. For more information on changing the recipients in the users listed in the "To:" line, see Usage Note “6” on page 574, "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 opt6 opt7

    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 NONOTEBOOK.

opt5
is the NOTEBOOK file name: ALL, *, or the file name specified in the NOTE command.

opt6
is CLASS.

Note: The CLASS option is handled differently from the other options in the OPTIONS: line. Because prior versions of the NOTE command did not have the CLASS option, the CLASS option does not have to appear in the OPTIONS: line.

opt7
is the spool class to use when sending the note.

Note: With the exception of the USEROPTIONS: line, if there are any lines between this options line and the Date: line, they must be blank.

The other header records are:

USEROPTIONS:
is an options line that can be used by user applications to list their unique options. The USEROPTIONS: line, if specified, must appear in the note file between the OPTIONS: line and the Date: line. If the SENDFILE command sends a note file that contains a USEROPTIONS: line, the information on the line is ignored and the USEROPTIONS: line is not sent as part of the note.

Note: The NOTE command will not create a USEROPTIONS: header record.

Date:
is the local date and time the note is prepared. The date and time are displayed as:

```
dd month yyyy, hh:mm:ss zzzzz
```

Where:

dd
specifies the day of the month.

month
specifies the name of the month.

yyyy
specifies the 4-digit year.

hh
specifies the hour (on a 24-hour clock).

mm
specifies the minutes.

ss
specifies the seconds.

zzzzz
specifies the offset from Coordinated Universal Time (UTC).

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.

13. 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, and so forth.

    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.

14. If you want to issue NOTE from an exec program, you should precede it with the EXEC command; that is, specify:

    ```
    exec note
    ```

15. In an SSI cluster, if you issue the NOTE command without specifying the node, and the recipient is a single-configuration virtual machine, the recipient can display or receive the note when logged on to any member of the cluster. If you omit the node and the recipient is a multiconfiguration virtual machine, the recipient can display or receive the note only when logged on to the member where the NOTE command was issued. If you specify the node for either type of user, the note will be sent only if RSCS is running on both members.

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.

For more information, see “Examples” on page 577. The information in the headings was collected from the names file, see the NAMEFIND command, “Examples” on page 532.

The command NOTE DWARFS CC: GORGEOUS where DWARFS and GORGEOUS are nicknames in the names file referenced above, produced the following screen:

```
 SNOWHITE NOTE     A0  V 132 Trunc=132 Size=24 Line=12 Col=1 Alt=0
OPTIONS: NOACK  LOG  SHORT  NOTEBOOK ALL  CLASS A




Date: 11 February 1996, 11:04:52 -0400
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 38. 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  CLASS A
Date: 11 February 1996, 11:04:52 -0400
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           111-1111            PRINCE   at ATLARGE
```

Figure 39. Example of Long Headings

Responses

Note canceled.

Messages and Return Codes

- DMS006E No read/write filemode accessed [RC=36]
- DMS149E Userid *userid* not valid; check your *userid* NAMES file [RC=32]
- DMS399E Tag too long for *nickname* in *userid* NAMES file [RC=88]
- DMS637E Missing {value|nodeid} for the {*option* option |*operand* operand} [RC=24]
- DMS647E Userid not specified for *nickname* in *userid* NAMES file [RC=32]
- DMS648E Userid *name* not {found|resolved}; check the *userid* NAMES file [RC=32]
- DMS651E ADD must be issued from NOTE [RC=40]
- DMS651E CANCEL must be issued from NOTE [RC=40]
- DMS653E Error executing *command*, rc=*rc* [RC=40]
- DMS665E File *userid* NOTE * not found; to begin a new note enter NOTE name [RC=28]
- DMS666E NOTE already exists; enter NOTE to continue, or specify REPLACE option [RC=28]
- DMS669E List of addressees cannot begin with CC: [RC=24]
- DMS1012E Node ID *node* not valid; check your *userid* NAMES file [RC=32]

Messages when in the NOTE environment (in XEDIT):

Note: Several of the above messages are also issued in the XEDIT environment.

- DMS579E Records truncated to lrecl when added to fn NOTEBOOK fm [RC=3]
- DMS667E Note header does not contain the {keyword From:|keyword To:|OPTIONS line|DATE line} [RC=32]
- DMS668E ADD option must be specified alone [RC=40]
- DMS670E No names to be added were specified [RC=24]
- DMS676E Invalid character * for Network ID [RC=20]
- DMS677E Invalid option *option* in option line [RC=32]
- DMS2501E One or more lines between the {OPTIONS:|USEROPTIONS:} line and the DATE: line contain non-blank characters.

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

NUCXDROP

Authorization

General User

Purpose

Use the NUCXDROP command to drop nucleus extensions. If the nucleus extensions are not in a segment the storage occupied by the corresponding program is released. For more information on the NUCXDROP command using the NUCEXT function, see *z/VM: CMS Macros and Functions Reference*.

Operands

name
Is the nucleus extension to be dropped.

*****
Means all currently loaded nucleus extensions. However, NUCXDROP * does not remove extensions loaded with the PERManent option of NUCXLOAD. They must be dropped by name.

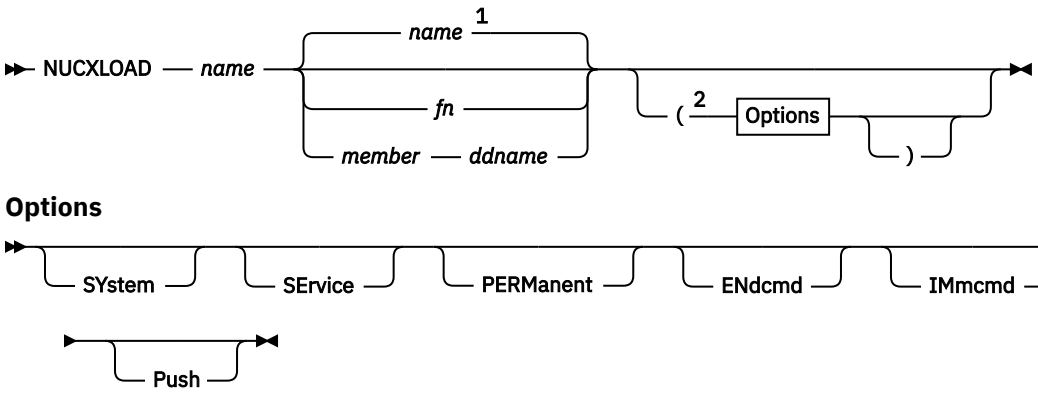
Messages and Return Codes

- DMS050E Parameter missing after NUCXDROP [RC=24]
- DMS070E Invalid argument *argument* [RC=24]
- DMS616W *name* does not exist [RC=28]
- DMS617E Error code *nn* from DMSFRET while unloading *module* module [RC=3]
- DMS624W No nucleus extensions were dropped [RC=4]
- DMS624W No nucleus extensions are loaded [RC=28]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

NUCXLOAD



Notes:

[1] The default is the name you specify.
[2] Options entered between the parenthesis can be in any order.

Authorization

General User

Purpose

Use the NUCXLOAD command 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 generated by the LOAD command with the RLDSAVE option, and then followed by the GENMOD command or a module generated by the BIND command.

These modules must be serially reusable modules. The nucleus extension is invoked by issuing the name of the nucleus extension. For more information on the NUCXLOAD command using the NUCEXT function, see *z/VM: CMS Macros and Functions Reference*.

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 the program needs no relocation, for example, it runs correctly when loaded at an address different from that at which it was generated (by 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 (for example, 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, or by using the BIND command.

member
must be a member of a CMS or OS load library. For more information on how to create a CMS load library, see "LKED" on page 465.

ddname
is the ddname from the FILEDEF command that must be issued before calling the NUCXLOAD that identifies the load library.

Options

SYstem
indicates 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 service calls are accepted (for instance a PURGE from an abend).

PERManent
indicates the extension cannot be dropped by a NUCXDROP * command; it must be named explicitly on NUCXDROP.

ENdcmd
indicates the nucleus extension receives control at normal end-of-command processing.

IMmcmd
indicates 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 canceled, either explicitly or implicitly. Implicit cancelation usually occurs only for nucleus extensions of the 'user' type (during an abnormal end cleanup time when all storage of 'user' type is reclaimed). Explicit cancelation does not release the storage (if any) occupied by the nucleus extension: that is the responsibility of the program that issues the cancelation (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. For more information, see the register contents in the NUCEXT macro in *z/VM: CMS Macros and Functions Reference*.
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 with the NUCXLOAD command or from a CMS module file that has relocation information saved. For more information about saving relocation information, see "INCLUDE" on page 417 and "LOAD" on page 471.
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'). For more information on the NUCEXT macro, see *z/VM: CMS Macros and Functions Reference* and *z/VM: CMS User's Guide*.

8. Not all high level programming languages can generate serially reusable object code. For example, OS/VS 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 processing.
10. When a nucleus extension is established by a multitasking application, it becomes associated with the process that created it, while also being known throughout the session. If a thread in another process invokes the nucleus extension, a thread is created in the process that established it to run the nucleus extension. In this way, it runs in the language environment of its process and if it abends, VMERROR event handlers established in that process can attempt recovery.
11. A program that is to be a nucleus extension must not be built with the multitasking initialization routine VMSTART. While a nucleus extension can perform multitasking operations, it cannot be the starting point for a new process.

    Note: For more information, see *z/VM: CMS Application Multitasking*.

Messages and Return Codes

- DMS001E No filename specified [RC=24]
- DMS070E Invalid parameter *parameter* [RC=24]
- DMS589E Missing FILEDEF for DDNAME SYSIN [RC=32]
- DMS618E NUCEXT failed [RC=*nn*]
- DMS619E Module *module* not found [RC=28]
- DMS622E Insufficient free storage [RC=*rc*]
- DMS639E Error in DMSRLD routine; return code was *nnn* [RC=24 | 31 | 55 | 70 | 76 | 99]
- DMS2521E NUCXLOAD cannot be performed on empty file *name* [RC=88]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411
Errors in loading a file	“Messages and Return Codes” on page 485

Return codes:

RC
Meaning

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 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 file name was not specified, an invalid operand was specified, or too many or extraneous operands were specified.

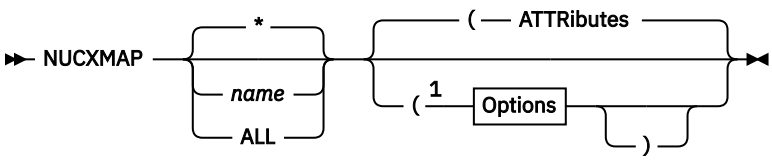
28
This is the return code generated by NUCXLOAD when the specified module cannot be found. It is also used in the case of an error when opening a LOADLIB file, in which case message DMSSOP036E is produced by the open routine.

32
For NUCXLOAD, a FILEDEF command identifying the load library must be issued prior to calling NUCXLOAD.

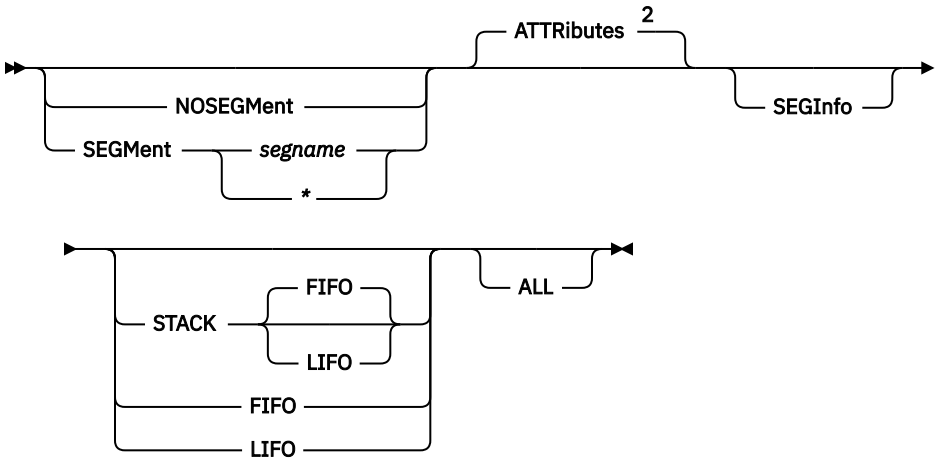
41
There was not enough free storage to build the table of SCBLOCK addresses.

100
An irrecoverable error occurred while reading the module from a disk or directory.

NUCXMAP



Options



Notes:

[1] Options entered between the parenthesis can be in any order.
[2] ATTRIBUTES is the default unless SEGINFO is specified. If SEGINFO is specified, attributes are not displayed unless ATTRIBUTES is also specified.

Authorization

General User

Purpose

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.

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 CMS displays information that describes each nucleus extension.

ALL
specifies CMS displays 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*
SEGMent *
returns information on nucleus extensions that reside in the specified loaded logical segment. The variable *segname* specifies the 1-8 character name of a logical segment. An asterisk (*) indicates 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 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. The header is not generated when the STACK option is specified.

FIFO
specifies 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 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 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
NUCGRNDL  004DB000 00000000 004DB000 00001FF8   24  SYSTEM
 PERM
```

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
```

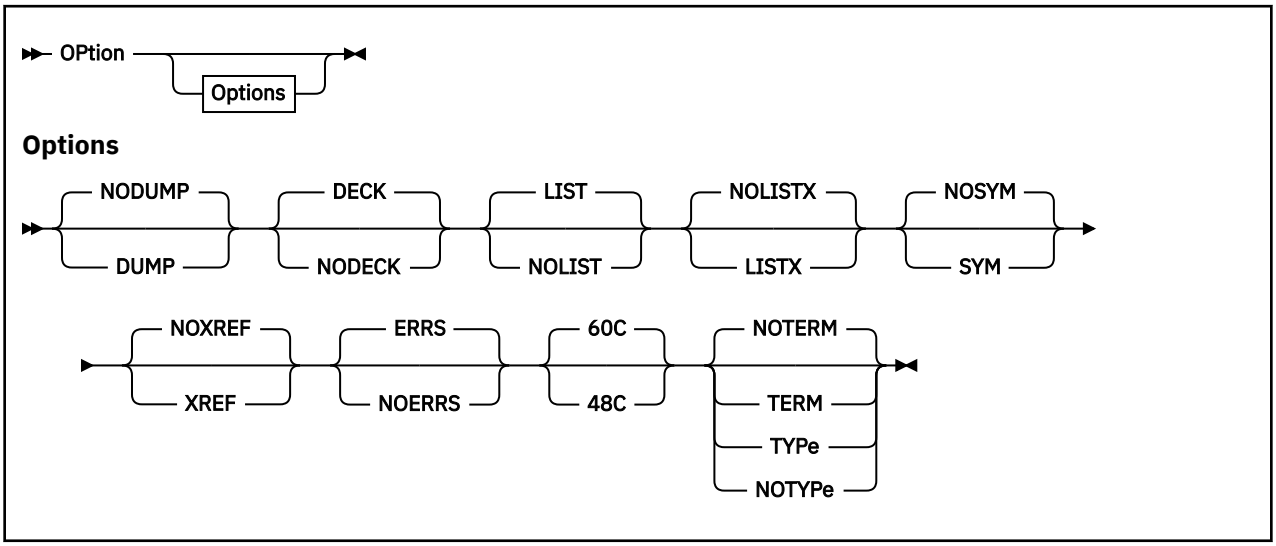
Messages and Return Codes

- DMS070E Invalid parameter *parameter* [RC=24]
- DMS622E Insufficient free storage [RC=*rc*]
- DMS624I No nucleus extensions are loaded
- DMS941I Nucleus extension *name* is not loaded

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

OPTION



Authorization

General User

Purpose

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.

Options

If a not valid 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. If you specify more than one option, CMS uses the last option specified.

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. This is the default.

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. This is the default.

NODECK
suppresses the DECK option.

LIST
writes the output listing of the source module on the SYSLST device. This is the default.

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. This is the default.

SYM
prints a Data Division map on SYSLST.

NOSYM
suppresses the SYM option. This is the default.

XREF
writes the output symbolic cross-reference list on SYSLST.

NOXREF
suppresses the XREF option. This is the default.

ERRS
writes an output listing of all errors in the source program on SYSLST. This is the default.

NOERRS
suppresses the ERRS option.

48C
Uses the 48-character set.

60C
Uses the 60-character set. This is the default.

TERM
displays all compiler messages to the user's terminal.

NOTERM
Suppresses the displaying of the compiler messages. This is the default.

TYPe
displays all compiler messages to the user's terminal. (This option has the same function as the TERM option.)

NOTYPe
suppresses the displaying of the compiler messages. (This option has the same function as the NOTERM 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 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

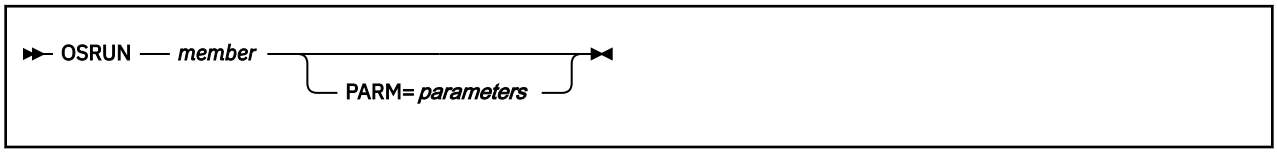
None.

To display a list of options currently in effect, use the QUERY command with the OPTION operand.

Messages and Return Codes

- DMS070E Invalid parameter *parameter* [RC=24]
- DMS099E CMS/DOS environment not active [RC=40]

OSRUN



Authorization

General User

Purpose

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.

Operands

member
is the member of a CMS LOADLIB or an OS module library to be executed.

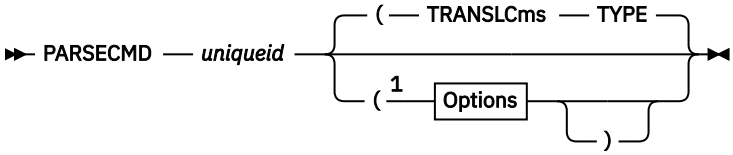
PARM=*parameter*
are the OS parameters the user wants to pass to the module. If the parameters contain blanks or special characters, they must be enclosed in quotation marks. To include quotations in the parameters, use double quotation marks. The parameters are passed in OS format: register 1 points 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 exec. The OSRUN command can be issued from a REXX or an EXEC 2 file with no restrictions.

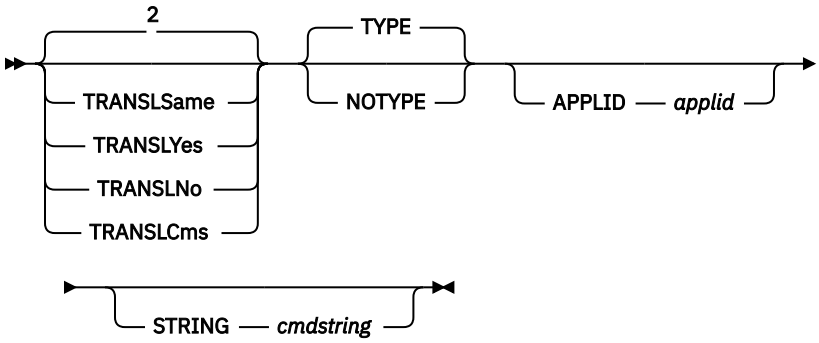
Messages and Return Codes

- DMS001E No filename specified [RC=24]
- DMS002I File(s) *fn* LOADLIB not found
- DMS013E Member *membername* not found in library *libname* [RC=32]
- DMS033E File *fileid* is not a library [RC=12]
- DMS052E More than 100 characters of options specified [RC=24]
- DMS070E Invalid parameter *parameter* [RC=24]
- DMS073E Unable to open file *ddname* [RC=28]

PARSECMD



Options



Notes:

[1] You can enter Options in any order between the parentheses.
[2] The default is TRANSLCms if the STRING option is not specified.

Authorization

General User

Purpose

Use the PARSECMD command to call the parsing facility from within an exec. The CMS parsing facility parses and translates command arguments.

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 16 characters in length.

Options

TRANSLSame
specifies the parsing facility should determine translation status from the first keyword found whose nl-name and sl-name DLCS definitions are different. In other words, the status is TRANSLNO if this keyword is the sl-name and TRANSLYES if this keyword is the nl-name. If all the nl-names and sl-names for the command are the same, the translation status cannot be determined.

The default is TRANSLSAME when STRING is specified. The translation status is returned from the *MESSAGE.1* variable if TRANSLSAME is explicitly specified and the parsing facility has successfully completed. *MESSAGE.1* will contain message 8160 with the string "TRANSLSAME" if a translation status cannot be determined.

TRANSLYes
specifies all keywords should be translated by the parsing facility. In other words, only keywords defined as nl-names in the DLCS syntax definition are recognized.

TRANSLNo
specifies keywords should not be translated by the parsing facility. In other words, only keywords defined as sl-names in the DLCS syntax definition are recognized.

TRANSLCms
specifies CMS determines translation status according to how the exec issuing PARSECMD was invoked. When CMS determines translation status, it uses:

- TRANSLYES if the specified exec name is a translation (or a synonym or abbreviation of a translation) of the exec invoked.
- TRANSLNO if the specified exec is a synonym (or an abbreviation of a synonym) set with the SYNONYM command of the exec invoked.
- TRANSLSAME if the exec is invoked using the same name by which the exec is specified.

For more information on how and when CMS translates or creates a synonym of a command name, For more information, see “SYNONYM” on page 1056. The default is TRANSLCMS when STRING is not specified. The translation status is returned in the *MESSAGE.1* variable if TRANSLCMS is explicitly specified and the parsing facility has successfully completed. *MESSAGE.1* will contain message 8160 with the string TRANSLYES or TRANSLNO if a translation status can be determined. It will contain message 8160 with the string TRANSLSAME if a translation status cannot be determined.

TYPE
displays error messages at the terminal for syntax errors that are found while parsing the exec arguments or *cmdstring*. The default is TYPE.

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 usually 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:

ALPHANUM
Alphanumeric string

APPLID
Any three character alphanumeric string with first alphabetic

ARBMODIF
Arbitrary modifier consisting of any string not defined by command syntax

COMMAND
Command name

COMMENT
Comment (everything following OPTEND)

CHAR
A single character

CSLPATH
The path number for a loaded CSL routine. It is a string consisting of two substrings separated by a period. Each substring is either an unsigned integer or an asterisk.

CUU
A 3-digit device address

DIGITS
Any unsigned number made up of digits 0-9

EFN
File name with '*' or '%' valid also

EFT
File type with '*' or '%' valid also

EXECNAME
Executable program name

EXECTYPE
Executable program type

FN
File name

FT
File type

FM
File mode

FPOOLID
File pool name, less than or equal to nine characters, with the first alphabetic, and the rest alphanumeric

HEX
Hexadecimal number

INTEGER
Integer: ..., -2, -1, 0, +1,+ 2, ...

KEYWORD
Keyword

MODE
Uppercase alphabetic character

NAMEDEF
A name definition, 16 characters or less, with the first character alphabetic, and the rest alphanumeric

NINTEGER
Negative integer: ..., -2, -1

OPTEND
Option end)

OPTSTART
Option start (

PINTEGER
Positive integer: +1, +2, ...

PN
Path name

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.

VDEV
A 4-digit device address

The validation codes that can be returned for the DIRID validation routine in the DLCS are:

DIRID
fully-qualified directory name, with file pool ID, user ID, and subdirectory name

DIRPOOL
directory name specified with file pool ID and subdirectory name

DIRUSER
directory name specified with user ID and subdirectory name

DIRSUB
subdirectory name

DIRMINUS
directory name specified with minus sign notation, for example: `-fm.subdirectory`.

DIRPLUS
directory name specified with plus sign notation, for example: `+fm.subdirectory`.

The validation codes that can be returned for the DIRIDN validation routine in the DLCS are:

FDIRNICK
full directory name, with file pool ID and nickname

DIRPOOL
directory name specified with file pool ID and subdirectory name

DIRNICK
directory name specified with nickname and user ID

DIRSUB
subdirectory name

DIRMINUS
directory name specified with minus sign notation, for example: `-fm.subdirectory`.

DIRPLUS
directory name specified with plus sign notation, for example: `+fm.subdirectory`.

3. The TYPE and NOTYPE options apply while parsing the syntax of *cmdstring* or EXEC arguments. Output with the TYPE option depends on the EMSG setting of the CP SET command. With the NOTYPE option, the complete message is copied into the variable *MESSAGE*, with the first line in *MESSAGE.1*, the second line in *MESSAGE.2*, and so forth.

   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 more information on unique IDs, see *z/VM: CMS Application Development Guide*.
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.
7. The parser will do translation for TRANSLCMS only when called from an exec invoked from the command line or with the CMS subcommand environment (ADDRESS CMS). The translation is assumed to have been done by the caller when the parser is invoked from an exec that was invoked CMSCALL (ADDRESS COMMAND).

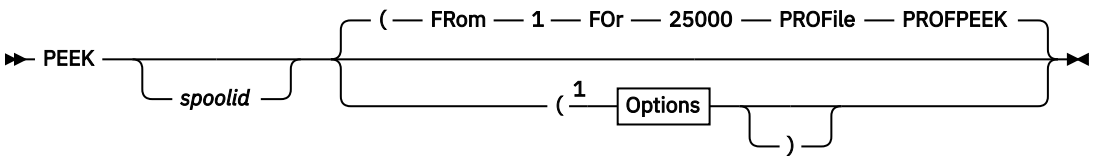
Messages and Return Codes

- DMS407E Invalid unique ID *uniqueid* [RC=24]
- DMS622E Insufficient free storage [RC=104]
- DMS631E *cmdname* can only be executed from an EXEC-2 or REXX EXEC [RC=40]
- DMS639E Error in *routine* routine; return code was *retcode* [RC=256]

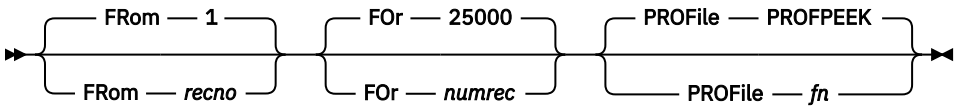
Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

PEEK



Options



Notes:

[1] You can enter options in any order between the parentheses.
A special command that can be used in the PEEK environment is DISCARD.

Authorization

General User

Purpose

Use the PEEK command to display a file 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.

Operands

spoolid
is the spool ID 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 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 25000 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 Usage Note “6” on page 596, "PF Key Settings on the PEEK Screen".

Usage Notes

1. For more information on how to customize this command see Appendix A, “Customizing Profiles for CMS Productivity Aids,” on page 1419.

2. IF PEEK is issued against a file printed to a VAFP printer, you are only allowed to see the first 204 characters of each line in the file.
3. You can use the special command DISCARD from the PEEK screen. The DISCARD command allows you to purge the reader file displayed by PEEK.
4. Tailoring the PEEK Command Options

   You can use the DEFAULTS command to set up options and to override the command defaults for PEEK. However, any option you specify on the PEEK command will override those specified in DEFAULTS. For more information, see "DEFAULTS" on page 153.
5. 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.
6. 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:

   PF1 Help
   Display PEEK command description.

   PF2 Add line
   Add a blank line after the current line.

   PF3 Quit
   Exit from the PEEK display.

   PF4 Tab
   Tab the cursor.

   PF5 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, see *z/VM: XEDIT Commands and Macros Reference*.

   PF6 ?/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 with PF5.)

   PF7 Backward
   Scroll backward one screen.

   PF8 Forward
   Scroll forward one screen.

   PF9 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.

   PF10 Rgtleft
   Shift the view to the right; press again to shift back to original display.

   PF11 Spltjoin
   Split a line or join two lines, at the cursor.

   PF12 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. To override these settings in a

profile, these SET subcommands must be stacked FIFO. Also, you should not use MAKEBUFs or DROPBUFs if stacking subcommands in the profile. This is because PEEK stacks subcommands on certain buffer levels and relies on these levels. A file is empty while any PEEK profile is running, and you cannot do anything to the data until the PEEK profile completes.

7. Processing Files You are PEEKing

   To process data in the file area of the file you are PEEKing, you must stack FIFO in your PEEK profile the name of an Xedit macro that will do the processing. Example PEEK profile:

```
    MYPEEK XEDIT A1  F  80  Trunc=80  Size=32

00000 * * * Top of File * * *
00001 /* */
00002 'QUEUE PROCFILE'
00003 Exit
00004 * * * End of File * * *
```

Figure 40. Xedit Macro Stacked in Your PEEK Profile

The following figure shows PROCFILE XEDIT, your Xedit macro that processes the data in your PEEK session's file area. This macro reads the first five lines from your PEEK edit session and displays them.

```
 PROCFILE  XEDIT  A1  F  80 Trunc=80 Size=32

00000 * * * Top of File * * *
00001 /* */
00002 'COMMAND :0'
00003  do 5
00004    'COMMAND +1'
00005    'COMMAND EXTRACT /CURLINE/'
00006    say curline.3
00007  end
00008 'COMMAND QQUIT'
00009 exit
00010 * * * End of File * * *
```

Figure 41. Sample Xedit Macro to Process Data in a PEEK Session

8. Files in DISK DUMP or NETDATA Format

   Files in DISK DUMP or NETDATA format are reformatted so they are readable.

   Note: If the multiple files are sent with continuous spooling (using CP SPOOL PUNCH CONT) and a series of DISK DUMP commands, PEEK will recognize only the first file in the continuous stack of files and a warning message will be shown.

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) files are being sent in this manner and list the file identifiers of the files you are sending.

9. Files in PUNCH Format

   If the punch is spooled continuous and PUNCH is used to send multiple files, those multiple files are treated as one file by PEEK and read in with ":READ" cards imbedded. In this case, the recipient must divide the file into individual files. This problem can be avoided by using SENDFILE or by not spooling the punch continuous.

   PEEK will not display the first record of a file generated using the PUNCH command with the HEADER option. This record contains the :READ header, as described in the PUNCH command.

10. 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.

11. Special NETDATA Files from MVS with TSO Extensions (PP)

    The MVS with TSO Extensions licensed program (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 the file is empty. Peeking at two files sent in one transmission results in two messages, identifying each of the files. You can also send a null file from an SFS directory, you will be able to PEEK it and get an empty file with no warning message. A line of equal signs (=) separates the two files.

12. The PEEK command does not handle MONITOR files or files with a SPECIAL status of YES. The SPECIAL status indicates whether the file contains records with X'5A' carriage control characters. For more information on how to determine the SPECIAL status of a file, see the CP QUERY command in *z/VM: CP Commands and Utilities Reference*.

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

    ```
    exec peek
    ```

14. In an SSI cluster, any spool file that resides in the shared spool of the cluster (except a file transmitted through RSCS) will appear to originate from the member where the PEEK command was issued. (For a file transmitted through RSCS, the actual origin node will be displayed.)

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= Rgtleft 11= Spltjoin 12= Cursor

====>
                                                              X E D I T  1 File
```

Figure 42. 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

- DMS132S File is too large [RC=88]
- DMS156E FROM *nnn* not found--the file has only *nnn* records [RC=32]
- DMS630S Error accessing spool file [RC=36]
- DMS643E No class *class* files in your reader [RC=28]
- DMS644E All reader files are in HOLD status or not class *class* [RC=28]
- DMS653E Error executing EXECIO [RC=*nn*]
- DMS655E Spoolid *nnnn* does not exist [RC=28]
- DMS672E Virtual reader invalid or not defined [RC=36]
- DMS674E Reader is not ready [RC=36]
- DMS683W The file has an LRECL of *nnn* and may have been truncated. [RC=32]
- DMS684E File contains invalid records and cannot be reformatted [RC=32]
- DMS687E This is a {SYSTEM{HELD | DUMP | LOCK}file|file with a special format} and cannot be peeked| This file has a special format and cannot be peeked [RC=1|10]
- DMS728I All records not shown. Use "PEEK (FOR *" to show all records.
- DMS1068W File *fn ft* is too large; some lines may not be shown
- DMS1069W Multiple files in spool file; only first file shown

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

PIPE

Authorization

General User

Purpose

Use the PIPE command to invoke CMS Pipelines.

Operands

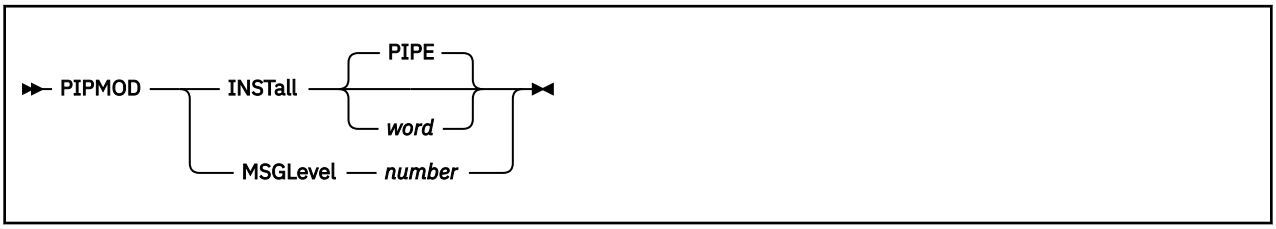
pipeSpec
is a pipeline specification that describes the pipeline to run with CMS Pipelines. The pipeline specification describes the individual pipeline programs to run, the arguments for these programs, and the way the input and output of these programs are connected.

A summary of the PIPE command is available in the HELP file. Enter:

```
help cms pipe
```

The detailed description of the pipeline specification, the options and built-in programs is provided in *z/VM: CMS Pipelines User's Guide and Reference*.

PIPMOD



Authorization

General User

Purpose

Use the PIPMOD command to set permanent CMS Pipelines options.

Operands

INSTall
creates the PIPE nucleus extension (if it does not already exist) and installs filter packages that are not already installed. If a word is specified with this operand, the word is used instead of PIPE as the name of the command that runs a pipeline specification.

MSGLevel *number*
sets the rightmost halfword of the pipeline message level to the specified number.

Usage Notes

For detailed information about the PIPMOD command, see *z/VM: CMS Pipelines User's Guide and Reference*.

PRELOAD

Authorization

Service Installer

Purpose

The PRELOAD command is a utility program that runs under CMS. It collects multiple text files and reformats them into a single text file. The function of the preloader is similar to that of a linkage editor, but the output is in standard text file format and does not include multiple CSECTS.

A program can be developed using separate assembly modules that reference each other. The preloader can then be used to combine the assembled text files into a single loadable text file.

Operands

loadlist
specifies the file name of an exec on the caller's A-disk or read-only extension containing records that define input to the preloader. Each of these records contains the file name and, optionally, the file type of an input text file. The format of each *loadlist* record that defines an input file must be:

```
&1 &2 filename  [filetype]
```

control
specifies the file name of a CNTRL file residing on one of the caller's accessed disks. The format and interpretation of the CNTRL file are the same as for the VMFLOAD command. It usually contains file types, sequenced by priority, to be used to select input files, if file types are not included in the *loadlist* file.

Note: PRELOAD ignores records that have a PTF update level identifier. It then searches for the next lower level identifier to determine the file type of the input text file. PRELOAD also ignores any options in the *loadlist*.

Input

The preloader gets input file names from the *loadlist*. The file type for each input file is determined in one of three ways:

1. If the *loadlist* record for a given input file includes a file type entry, that file type locates the record.
2. If the *loadlist* record does not contain a file type, and a 'control' parameter was specified on the PRELOAD command line, the file type constructed is in the format TXT*xxxxx*. In this case, *xxxxx* is the highest control level identifier in the control file for which a file can be located on the caller's accessed disks.
3. If no file type is specified on the *loadlist* entry, and a control file has not been specified on the PRELOAD command line, the default file type value is TEXT.

Note: Input files are located by scanning the caller's disks in their access order. All input files must be on accessed disks.

Output

The preloader output consists of two files written to the caller's A-disk: *fn* TEXT and *fn* MAP.

The file name for each of these files is the same as that specified for the input *loadlist* file. If either of these files already exists on the caller's A-disk, the new copy replaces the old one.

TEXT File

The output TEXT file is a merged and linked composite of the input files. The first CSECT or private code section in the input expands to contain all input files. Its length is the sum of the lengths of the input files, rounded up to doubleword multiples between sections. Input TXT records of nonzero length are relocated and written to the output file.

The output RLD is a translated and relocated collection of all input RLD records. No sorting is done by the preloader. In general, each output ESD, TXT, and RLD entry appears in the same order as the corresponding input entry. ADCON and VCON fields are relocated within their TXT records. ORG statements that cause relocatable constant fields to overlay or be overlaid may cause results that differ from results obtained with a loader that completes TXT data loading before relocating ADCONs and VCONS.

MAP File

The output MAP file is a printable record of preloader processing, similar to a load map. The first line of the map contains:

- Output text file name
- Residence volume label and volume device address
- Date and time of file creation

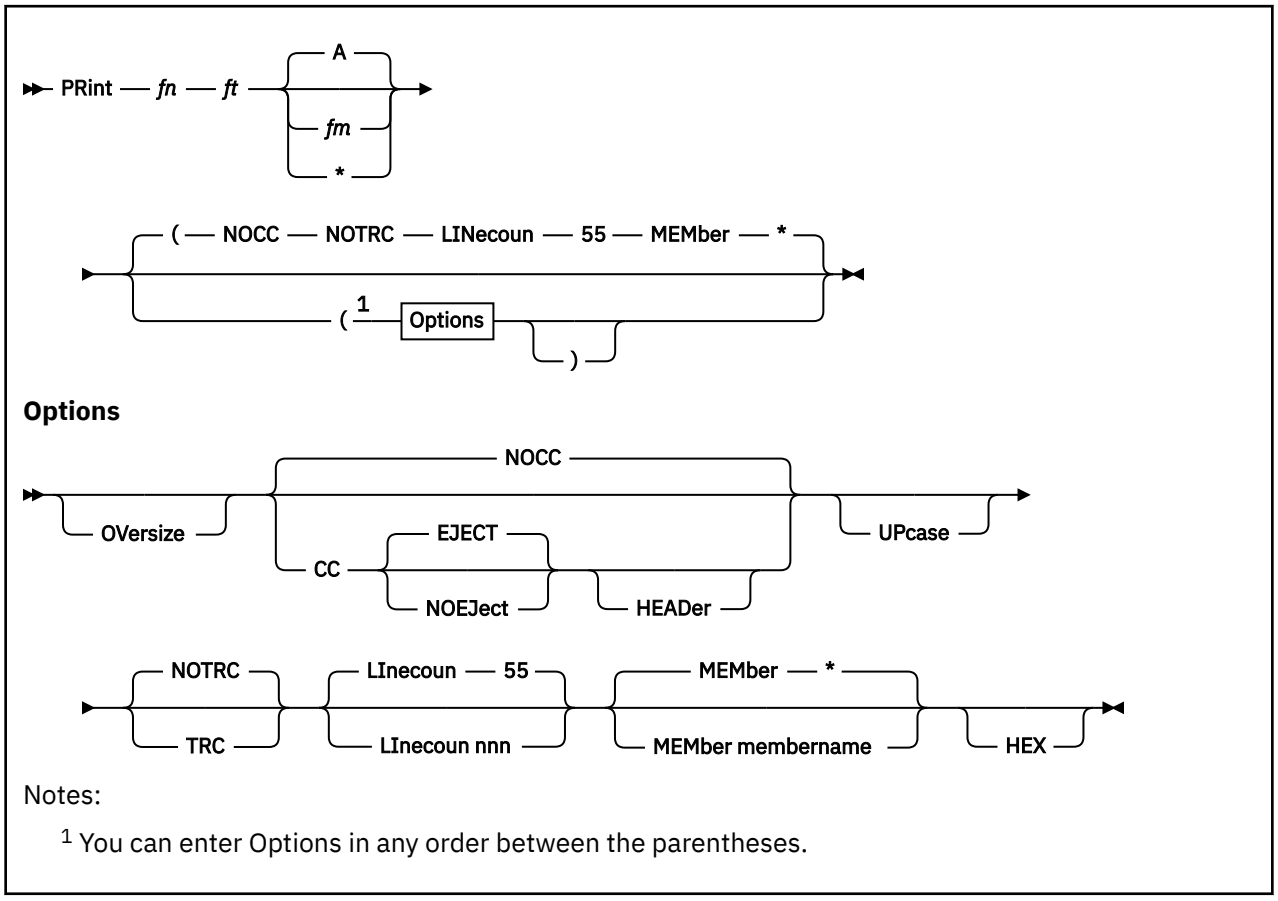
The next section of the map is a listing of the control file (if any) used. The remainder of the map contains, in processing order, a section for each input file. Each of these sections consists of:

- File name, file type, file mode of input file
- Residence volume label and virtual device number
- Input file's creation date and time
- Any not valid input records

Messages and Return Codes

- DMS001E No filename specified [RC=24]
- DMS002E File *fn ft fm* not found [RC=28]
- DMS104S Error *nn* reading file *fn ft fm* from disk or directory
- DMS105S Error *nn* writing file *fn ft fm* on disk or directory
- DMS109S Virtual storage capacity exceeded
- DMS179E Missing or invalid MACS card in control file *fn ft fm*
- DMS183E Invalid {CONTROL|AUX} file control card [RC=32]
- DMS234E Error in LOAD LIST file *fn ft fm* [: no input] [RC=8]
- DMS235E Error 5 on entry symbol *name* [RC=12]
- DMS236E Unresolved external reference(s) encountered [RC=4]
- DMS237E Duplicate external symbol(s) encountered [RC=8]
- DMS238E Preloader processing error [RC=16]

PRINT



Notes:

[1] You can enter Options in any order between the parentheses.

Authorization

General User

Purpose

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, see *z/VM: CP Planning and Administration* to set up the address.

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 with records larger than the carriage size of the virtual printer. Records larger than the carriage size of the virtual printer are printed, but are truncated to the carriage size (or carriage size + 1 if CC is specified). If, however, the record's carriage control character is X'5A', the record may be 32,767 bytes long. Generally, files containing X'5A' carriage control characters can only be printed on printers supported through the Print Services Facility™ (PSF), such as the IBM 3820 and IBM 3800 Model 3.

If the CC option is in effect and an X'5A' carriage control character is processed, the resulting printer spool file will have a SPECIAL status of YES; otherwise, it will have a SPECIAL status of NO. For more information on spool file SPECIAL status, see the QUERY PRINTER PSF command in *z/VM: CP Commands and Utilities Reference*.

If the OVERSIZE option is not in effect and the file you want to print is larger than the carriage size of the virtual printer, you will receive the message "Record length exceeds allowable maximum".

If the HEX option is specified, records up to 2,147,483,647 bytes long may be printed regardless of whether the OVERSIZE option is in effect.

Default: If the file type is LISTCDPS, LIST3820, or LIST38PP, OVERSIZE is assumed; otherwise, it is not.

CC
interprets the first character of each record as a carriage control character.

When CC is used and the printer file is spooled to the user's reader, the carriage control characters are not preserved if the file is used as input to any subsequent operation (like MOVEFILE).

The LINECOUN option is ignored if CC is in effect.

EJECT
causes a page eject before any records in the file are processed. This ensures printing begins at the top of a new page. The default is EJECT. If CC is not specified, EJECT is ignored.

NOEJECT
ensures no page eject is performed before the file is processed. Page ejects are controlled by the carriage control characters in the file. If CC is not specified, NOEJECT is ignored.

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. 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. The maximum page value in the header is 99999.

If NOCC is specified, it is in effect even if the CC or OVERSIZE option is specified or if the file type is LISTING, LIST3800, LISTCPDS, LIST3820, or LIST38PP.

Default: The default is CC if the HEX option is not specified and either the OVERSIZE option is specified, or the file type is LISTING, LIST3800, LIST3820, LIST38PP, or LISTCDPS; otherwise the default is NOCC.

UPcase
translates the lowercase letters in the file to uppercase for printing. If the carriage control character is X'5A', the record is not translated.

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 (with the CHARS keyword of

the SETPRT command). The valid values for TRC are 0, 1, 2, and 3. If a not valid 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. Default: TRC is the default if the HEX option is not specified and the file type is LIST3800; otherwise, NOTRC is the default.

LInecoun
allows you to set the number of lines (*nnn*) to be printed on each page. *nnn* can be any decimal number from 0 through 144. If the LINECOUN option is not specified, the default LINECOUN 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 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). For more information, see Usage Note "8" on page 607.

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. This is the default. 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, HEADER, TRC, and UPCASE are ignored, even if specified, and even if the file type is LISTING, LIST3800, LISTCPDS, LIST3820, or LIST38PP.

Usage Notes

1. The file may contain carriage control characters and may have either fixed- or variable-length records. The maximum number of characters in a record is:

Virtual Printer Type	Maximum Characters
1403	132
3203	132
3800	204
4248	168
VAFP	32767

There are exceptions, if the:

- CC option is in effect, the record length can be one character longer (133 or 169) to allow for the carriage control character.
- 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.
- HEX option is in effect, a record of any length can be printed, up to the CMS file system maximum of 2,147,483,647 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 unless the OVERSIZE option is specified or the file type is LISTING, LIST3800, LIST3820, LIST38PP, or LISTCPDS. 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. The valid carriage control codes are:

Character	Meaning
blank	Space 1 line before printing

Character	Meaning
0	Space 2 lines before printing
-	Space 3 lines before printing
n, where *n* is 1-9	Skip to channel *n*
A	Skip to channel 10
B	Skip to channel 11
C	Skip to channel 12

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 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 *
```

   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.
7. You may not be authorized to print files on all printers if an external security manager (ESM) is installed on your system and security label checking is enabled. Your files may not be printed and message HCP356E, HCP1561E, or HCP2514E issued to the system operator if you are not authorized to print files on the printer you selected. For additional information, contact your security administrator.
8. If the value of the LPP option on the CP SPOOL command is 0 (LPP OFF), the default LINECOUN value is 55. If the LPP value is greater then 0, the LINECOUN value will be set to the LPP value minus 3 to account for the extra header lines.

   If the LINECOUN option is specified with the command, the value of LINECOUN overrides the value set by the LPP option. For more information, see *z/VM: CP Commands and Utilities Reference*.

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.

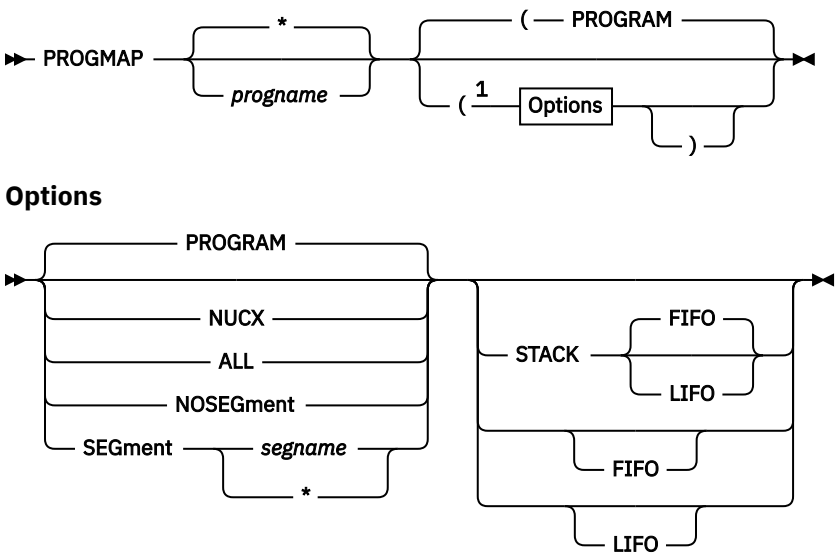
Messages and Return Codes

- DMS002E File [*fn* [*ft* [*fm*]]] not found [RC=28]
- DMS008E Device *vdev* {invalid or nonexistent|is an unsupported device type} [RC=36]
- DMS013E Member *membername* not found in library *libname* [RC=32]
- DMS029E Invalid parameter *parameter* [in the [option] *option* field] [RC=24]
- DMS033E File *fn ft fm* is not a library [RC=32]
- DMS039E No entries in library *fn ft fm* [RC=32]
- DMS044E Record exceeds allowable maximum [RC=32]
- DMS069E Filemode *mode* not accessed [RC=36]
- DMS104S Error *nn* reading file *fn ft fm* from disk or directory [RC=100]
- DMS109S Insufficient free storage available [RC=104]
- DMS123S Error *nn* printing file *fn ft fm* [RC=100]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411
Errors in the Shared File System	“File Pool Server Messages” on page 1414

PROGMAP



Notes:

[1] Options entered between the left '(' and right ')' delimiters can be in any order.

Authorization

General User

Purpose

Use the PROGMAP command to display or place on the program stack information about programs currently loaded in storage.

Operands

progname
returns information about all programs with the specified program name.

- For a program loaded by the LOAD, INCLUDE, or LOADMOD commands, OS Simulation (LOAD, LINK, or ATTACH), or MODULEs invoked as commands (that have NOCLEAN attribute), *progname* is the file name.
- For a nucleus extension, *progname* is the command name.

An asterisk (*) will return information on all user programs, or nucleus extensions, or both. The default is an asterisk.

Options

PROGRAM
returns information for program(s) loaded by the LOAD, INCLUDE, or LOADMOD commands, OS Simulation (LOAD, LINK, or 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 all logical segments are to be searched for the nucleus extensions specified by *progname*.

STACK FIFO
STACK LIFO
specifies 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 more information on NUCEXT look-aside entries, see "NUCXMAP" on page 584.
3. PROGMAP does not display the attributes of transient area modules.

Examples

If you enter PROGMAP PROG1, only information for PROG1 will be returned:

```
Name        Entry       Origin      Bytes        Attributes
PROG1       02000400    04000400    0000066D    Amode 31  Reloc
```

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       00020000    00020000    0000C778    Amode 24 Non-reloc

Name  Entry    Userword Origin   Bytes   Amode (Attributes)
NUCX1 0035A000 00000000 00000000 00000000   31  SYSTEM SERVICE
NUCX2 0035E934 00361828 00000000 00000000  Any  SYSTEM
NUCX3 004DB000 00000000 004DB000 00001FF8   24  SYSTEM SERVICE IMMCMD PERM
```

Note: All text decks are considered to be nonrelocatable programs whether 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 the module file will be relocated when it is loaded by LOADMOD.

Messages and Return Codes

- DMS415E Invalid character *char* in program name *name* [RC = 20]
- DMS941I User program *progid* is not loaded [RC = 0]
- DMS942I No user programs are loaded [RC = 0]

An additional system message may be issued by this command. The reason for this message and its location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

PSCREEN PUT

Authorization

General User

Purpose

Use the PSCREEN PUT command to copy the image of a physical screen to a CMS file.

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.

Usage Notes

1. The PSCREEN PUT 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 first disk or directory accessed R/W (generally this is A). 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 the physical screen 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

- DMS037E Filemode *mode* is accessed as read/only [RC=12]
- DMS048E Invalid filemode *mode* [RC=24]
- DMS062E Invalid character *char* in fileid [*fn ft [fm]*] [RC=20]
- DMS069E Filemode *mode* not accessed [RC=36]
- DMS105S Error *nn* writing file *fn ft fm* on disk or directory [RC=100]
- DMS107S Disk *mode(vdev)* is full [RC=100]
- DMS386E Missing operand(s) [RC=24]
- DMS388E Invalid keyword: *keyword* [RC=24]
- DMS391E Unexpected operand(s): *operand* [RC=24]
- DMS531E Disk or file space is full; set new filemode or clear some disk space [RC=13]
- DMS917E No windows are displayed [RC=4]
- DMS924E Data was truncated [RC=3]
- DMS1144E Implicit rollback occurred for work unit *workunitid* [RC=31]

- DMS1252T Rollback unsuccessful for file pool *filepoolid*
- DMS1258E You are not authorized to write to file *fn ft fm* | *dirid* [RC=28]
- DMS1262S Error *nn* opening file *fn ft fm* [RC=100]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411
Errors in the Shared File System	“File Pool Server Messages” on page 1414

PSCREEN REFRESH

Authorization

General User

Purpose

Use the PSCREEN REFRESH command to update virtual screens and their associated windows and refresh the screen.

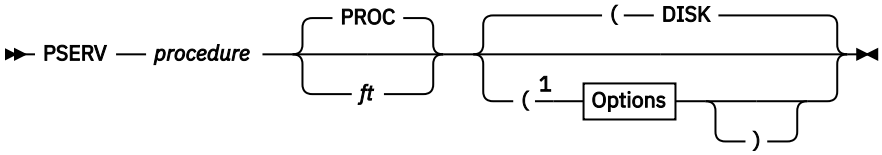
Messages and Return Codes

- DMS391E Unexpected operand(s): *operand* [RC=24]
- DMS622E Insufficient free storage [RC=104]
- DMS917E No windows are displayed [RC=4]
- DMS925E I/O error on screen [RC=100]
- DMS926E Command is only valid on a display terminal [RC=88]

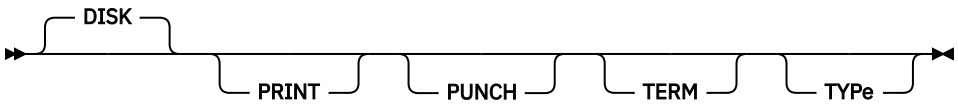
Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

PSERV



Options



Notes:

[1] You can enter Options in any order between the parentheses.

Authorization

General User

Purpose

Use the PSERV command in CMS/DOS to copy, display, print, or punch a procedure from the VSE procedure library.

Operands

procedure
specifies the name of the procedure in the VSE procedure library 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 choose, depending on the functions you want to perform. If you specify more than one option, CMS uses the last option specified.

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.

TYPe
displays the procedure on your terminal. (This option has the same function as the TERM option.)

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.

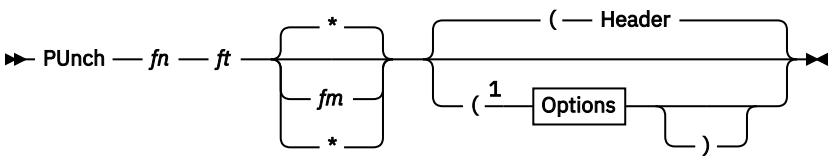
Responses

When you issue the TERM option, the procedure is displayed at your terminal.

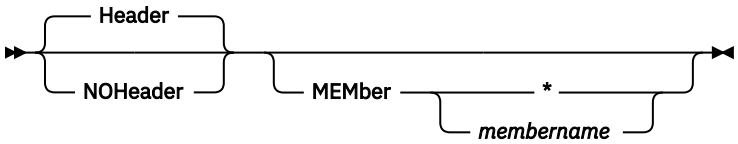
Messages and Return Codes

- DMS003E Invalid option: *option* [RC=24]
- DMS004E Procedure *procedure* not found [RC=28]
- DMS006E No read/write A filemode accessed [RC=36]
- DMS070E Invalid parameter *parameter* [RC=24]
- DMS097E No SYSRES volume active [RC=36]
- DMS098E No procedure name specified [RC=24]
- DMS099E CMS/DOS environment not active [RC=40]
- DMS105S Error *nn* writing file *fn ft fm* on disk or directory [RC=100]
- DMS113S Disk(*vdev*) not attached [RC=100]
- DMS411S Input error code *nn* on SYSRES [RC=*rc*]

PUNCH



Options



Notes:

[1] You can enter Options in any order between the parentheses.

Authorization

General User

Purpose

Use the PUNCH command to punch a CMS file to your virtual punch.

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 is the default. 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 38 on page 618.

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 38. 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	*fname*	File name of the file punched.
16	1	blank	
17-24	8	*ftype*	File type of the file punched.
25	1	blank	
26-27	2	*fmode*	File mode of the file punched.
28	1	blank	
29-34	6	*volid*	Label of the disk from which the file was read or a "—" if read from a directory.
35	1	blank	
36-43	8	*mm*/*dd*/*yy*	The date the file was last written.
44-45	2	blank	
46-50	5	*hh:mm*	The time of day 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: 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) 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 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 the file is being spooled to a particular user's virtual reader.

When you PUNCH empty files, file attribute data is sent. You must use the HEADER option, or you will get this message:

```
Error punching file fileid; NOHEADER option invalid
for empty files
```

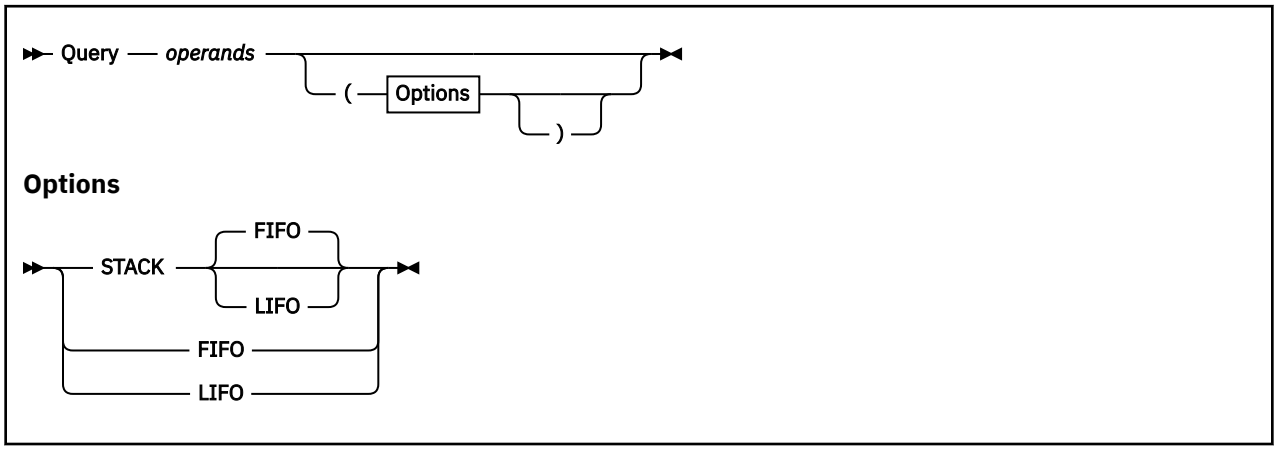
Messages and Return Codes

- DMS002E File [*fn* [*ft* [*fm*]]] not found [RC=28]
- DMS008E Device *vdev* {invalid or nonexistent|is an unsupported device type} [RC=36]
- DMS013E Member *membername* not found [RC=32]
- DMS026E Invalid parameter *parameter* in the option field *fieldname* [RC=24]
- DMS033E File *fn ft fm* is not a library [RC=32]
- DMS039E No entries in library *fn ft fm* [RC=32]
- DMS044E Record exceeds allowable maximum [RC=32]
- DMS069E Filemode *mode* not accessed [RC=36]
- DMS104S Error *nn* reading file *fn ft fm* from disk or directory [RC=100]
- DMS118E Error punching file *fileid*; NOHEADER option invalid for empty files [RC=24]
- DMS123S Error *nn* punching file *fn ft fm* [RC=100]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

QUERY



Authorization

General User

Purpose

Use the QUERY command to gather information about your CMS virtual machine. You can get information about:

- The operation of your virtual machine, such as:
  - The status of virtual machine characteristics that are controlled by the CMS SET command
  - The status of CMS/DOS functions
  - The search order for libraries (MACLIBs, TXTLIBs, CSLLIBs, DOSLIBs, and LOADLIBs)
  - Information about your saved segments.
- The status of your files and file pool directories, such as:
  - File definitions (set with the FILEDEF and DLBL commands) that are in effect
  - The status of your accessed disks, and file pool and Shared File System (SFS) directories
  - Authorities that have been granted on file pools and SFS directories or the files in them
  - Aliases that exist on files in file pools and SFS directories
  - The value of recoverability and overwrite file attributes in file pools
  - Information about your file pool
  - Locks existing on file pools and SFS directories or the files in them.
  - Disable lock information about a file pool file space or its owning storage group.
- Information about how your virtual machine is set up for windowing in full-screen CMS, such as:
  - Characteristics of your physical screen
  - CMSPF and WMPF key settings
  - Characteristics of windows you have defined, and the order in which the windows are being displayed
  - Characteristics of the virtual screens which support the windows.

The following are the operands available with QUERY:

ABBREV
ACCESSED
ACCESSM0
ACCESSORs
ALIAS
APL
AUThority
AUTODUMP
AUTOREAD
BLIP
BLOCKS
BORDER
CHARMODE
CMSLEVEL
CMSPF
CMSREL
CMSTYPE
CMS370AC
COMDIR
CONNect
CSLLIB
CURSOR
DIRATTR
DISK
DISPLAY
DLBL
DOS
DOSLIB
DOSLNCNT
DOSPART
ENROLL
ETRACE
EXECTRACe
FILEATTR
FILEDEf
FILEPOOL
FILEPOOL CONFLict
FILEPOOL CURRENT
FILEPOOL
DISable
FILEPOOL PRIMARY
FILESPACE
FILEWAIT
FULLREAD
FULLSCREen
GEN370
GETMAIN
HIDE
IMESCAPE
IMPCP
IMPEX
INPUT
INSTSEG
KEY
KEYPROTect
LABELDEF
LANGLIST
LANGuage
LDRTBLS
LIBRARY
LIMITS
LINEND
LOADAREA
LOADLIB
LOCATION
LOCK
LOGFILE
MACLIB
MACLsubs
NAMEDEF
NONDISP
OLDCMDS
OPTION
OSTXTBUF
OUTPUT
PROTECT
RDYMSG
RECALL
REDTYPE
RELPAGE
REMOTE
RESERVED
RORESPECt
ROUTE
SEARCH
SEGMENT
SERVER
SHOW
STORECLR
SYNONYM
SYSNAMES
TAPECSL
TAPENEVR
TEXT
TRACECTL
TRANslate
TRAPMSG
TVICALL
TXTLIB
UPSI
VSCREEN
WINDOW
WMPF

Options

General

The following options apply with all operands of the QUERY command:

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 a QUERY operand is unknown to CMS, CMS will pass it to CP rather than issue a -3 return code. The response from CP is then put in the program stack. If CP precedes the QUERY command, CMS does not stack the results. 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, 165 for QUERY AUTHORITY, or 49 for QUERY FILEATTR.

The XEDIT option of the QUERY command can only be used with QUERY ALIAS, QUERY AUTHORITY, and QUERY FILEATTR.

Usage Notes

General

These notes apply with all operands of the QUERY command:

1. You may specify only one QUERY operand at a time.
2. If the implied CP (IMPCP) function is in effect and you enter a not valid QUERY operand, you may receive the message HCPCQG045E - userid NOT LOGGED ON.
3. If a not valid QUERY operand is specified from an exec, the implied CP (IMPCP) function is in effect, and no stack options are specified, then the return code is -3 and nothing is placed in the program stack.
4. When the STACK option is specified, the header is included in the program stack. If the information you query requires a response from CP and the response is longer than 8192 characters, nothing is stacked and you receive message DMSQRY627E and a return code of 88.
5. The language for QUERY command responses depends on the language used to enter the command and whether the command was translated. If the QUERY command is entered in American English (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 "Using Translations" in *z/VM: CMS User's Guide*.

Messages and Return Codes

These error messages may be returned by any of the QUERY operands.

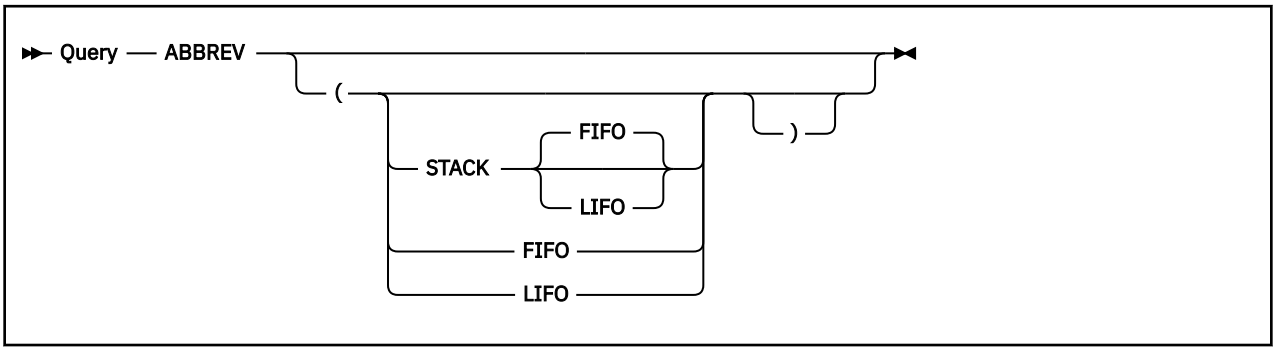
- DMS002E File *fn ft fm* | *dirname* not found [RC=28]
- DMS003E Invalid option: *option* [RC=24]
- DMS006E No read/write {disk|filemode|*filemode* filemode} accessed [for *fn ft*]
- DMS014E Invalid function *function* [RC=24]
- DMS065E *option* option specified twice [RC=24]
- DMS066E *option1* and *option2* are conflicting options [RC=24]
- DMS069E {[Output]{filemode|disk}*mode* [(*vdev*)]|Directory *dirname*} not accessed [RC=28]
- DMS070E Invalid parameter *parameter* [RC=24]
- DMS099E CMS/DOS environment [not] active [RC=40]
- DMS104S Error *nn* reading SYSTEM LANGUAGE S from disk [RC=1nn]
- DMS105S Error *nn* writing file to XEDIT [RC=100]
- DMS109S Virtual storage capacity exceeded [RC=109]
- DMS149E Userid *userid* not valid [RC=32]

- DMS280E Application *applid* not active
- DMS386E Missing operand(s) [RC=24]
- DMS389E Invalid filepoolid: *filepoolid* [RC=24]
- DMS391E Unexpected operands(s): *operand* [RC=24]
- DMS525E Invalid PF Key number [RC=24]
- DMS618E NUCEXT failed [RC=*nn*]
- DMS621E Bad plist: *message* [RC=24]
- DMS627E Result is *nnn* bytes too large for CP command buffer [RC=88]
- DMS637E Missing nodeid for the AT operand [RC=24]
- DMS639E Error in NUCXLOAD routine; return code was *nnn* [RC=24]
- DMS639E Error in *routine* routine; return code was *nnnn*
- DMS647E Userid not specified for *nickname* in *userid* NAMES file [RC=32]
- DMS647E Localid not specified for *userid* at *node* in *userid* NAMES file [RC=32]
- DMS653E Error executing *command* rc=*nn* [RC=40]
- DMS688E XEDIT option only valid from XEDIT environment [RC=24]
- DMS689E File must be F-format *nnn* or V-format [RC=24]
- DMS918E No {virtual screens|windows} are defined [RC=4]
- DMS916E Window *wname* is not {displayed|hidden} [RC=4]
- DMS917E No windows are {displayed|hidden} [RC=4]
- DMS921E Virtual screen *vname* is not defined [RC=28]
- DMS926E Command is only valid {in CMS FULLSCREEN MODE|on a display terminal} [RC=88]
- DMS1082E No window qualifies as the window on top [RC=4]
- DMS1184E File *fn ft fm* | *dirname* not found or you are not authorized for it [RC=28]
- DMS1184E Directory *dirname* not found or you are not authorized for it [RC=28]
- DMS1184E File *fn ft* or directory *dirname* not found or you are not authorized for it [RC=28]
- DMS1184E File *fn ft* or directory *dirid* not found [RC=28]
- DMS1188E File mode *fm* not associated with a directory. [RC=74]
- DMS1209E Nickname *nickname* resolved to more than one userid; query can be performed only on one userid at a time [RC=88]
- DMS1210E Directory *dirname* not found [RC=28]
- DMS1222I No user defined NAMEDEF in effect
- DMS1223E There is no default file pool currently defined [RC=40]
- DMS1239E You are not authorized to issue this request on behalf of *userid* [RC=76]
- DMS1239E You are not authorized to issue this request for GROUP or ALL [RC=76]
- DMS2023E File pool *filepool id* does not support the requested QUERY command.
- DMS3711E User *userid* not enrolled in the file pool *filepoolid* [RC=40]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411
Errors in the Shared File System	“File Pool Server Messages” on page 1414

QUERY ABBREV



Authorization

General User

Purpose

Use the QUERY ABBREV command to display the status of the minimum truncation indicator. For more information on how to change the setting of the indicator, see "SET ABBREV" on page 905.

Responses

```
ABBREV =   ON

or

ABBREV =   OFF
```

Where:

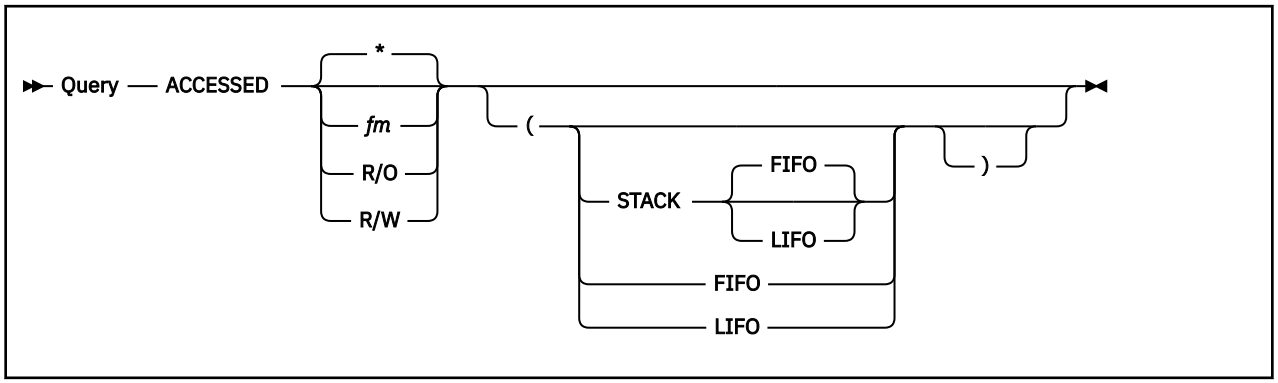
ON
indicates truncations are accepted for CMS commands and all translations.

OFF
indicates truncations are not accepted.

Options

For more information on the Options, Usage Notes and Error Messages that apply to all operands of the QUERY command, see "QUERY" on page 620.

QUERY ACCESSED



Authorization

General User

Purpose

Use the QUERY ACCESSED command to display the status of your accessed disks and SFS directories.

Operands

fm
displays status of a single disk or Shared File System (SFS) directory represented by *fm*.

*****
displays status of all your accessed disks or SFS directories. This is the default.

R/O
displays status of your accessed read-only CMS disks or SFS directories.

R/W
displays status of your accessed read/write CMS disks or SFS directories.

Response 1

QUERY ACCESSED displays one line for each accessed disk of the type specified.

```
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. If the directory has the directory control (DIRCONTROL) attribute, 'DIRC' is displayed instead of 'DIR'.

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.

Examples

The following is an example of the results of QUERY ACCESSED:

```
Mode  Stat     Files  Vdev  Label/Directory
A      R/W      1325  191   SMI191
B/B    R/O         0  18F1  OSTEST   - OS
C      R/W         0  300   VSAM01   - DOS
D      R/W        31  DIR   FILEPOOL:SMITH.
E/A    R/O      1281  5FF   CP5FF
G      R/O      4760  DIRC  FILEPOOL:TOOLS.GOODIES
S      R/O       350  190   ESA190
Y/S    R/O       903  19E   ESA19E
```

Response 2

If the disk or directory with the specified file mode, or of the specified access type, is not accessed, the possible responses are:

```
Disk fm not accessed

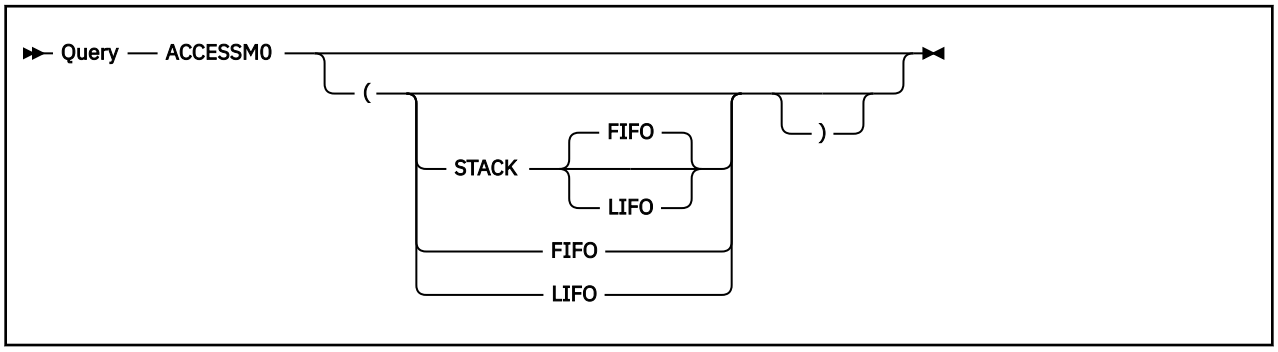
No filemode is read-only

No read/write disk accessed
```

Options

For more information on the Options, Usage Notes and Error Messages that apply to all operands of the QUERY command, see "QUERY" on page 620.

QUERY ACCESSM0



Authorization

General User

Purpose

Use the QUERY ACCESSM0 command to determine whether file mode 0 files can be brought into storage if the MODE0 option is specified or implied on the ACCESS command. Use the ACCESSM0 command to change this setting. For more information, see “ACCESSM0” on page 41.

Responses

```
ACCESSM0 ON
```

or

```
ACCESSM0 OFF
```

Where:

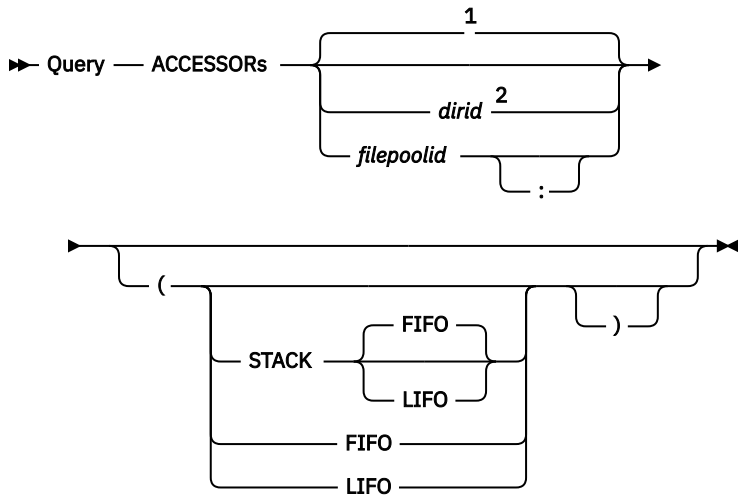
ON
accessing of file mode 0 files is enabled.

OFF
accessing of file mode 0 files is not enabled.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see “QUERY” on page 620.

QUERY ACCESSORS



Notes:

[1] The default file pool is used.
[2] If *dirid* is not specified, information for all your directory control directories is displayed.

Authorization

General User

Purpose

Use the QUERY ACCESSORS command to display information about current accessors of directory control directories.

To use the command, you must either have administrator authority, own the directories, or specify a directory ID to which you have directory control write authority.

Operands

dirid
identifies a directory control directory for which you want to display information. If you omit *dirid*, information is displayed for all your directory control directories. You can display information for another user's directory only if you have directory write authority to the directory or file pool administration authority. You can also use a file mode for the *dirid* if the directory is already accessed. For a more information on the *dirid*, see “Naming Shared File System (SFS) Directories” on page 6.

filepoolid
filepoolid:
names the file pool for which the query is intended. The colon is optional. If you omit the directory ID, you can specify the file pool ID to establish the file pool for the query. If both *filepoolid* and *dirid* are omitted, the default file pool is used. The file pool ID should not be specified if the directory ID is specified because the directory ID includes the file pool ID.

Responses

QUERY ACCESSORS displays one line for each accessor of each directory specified. If information for more than one directory is displayed, the information is returned in alphabetic order by directory name.

```
Accessors  Mode  Directory Name

userid     mode  dirname
  .          .       .
  .          .       .
  .          .       .
```

Note: The header is generated only if output is displayed at the terminal.

Where:

userid
is the ID of the user who has the directory accessed.

mode
is the status of the SFS directory: R/O (read-only) or R/W (read/write).

dirname
is the name of the directory being accessed.

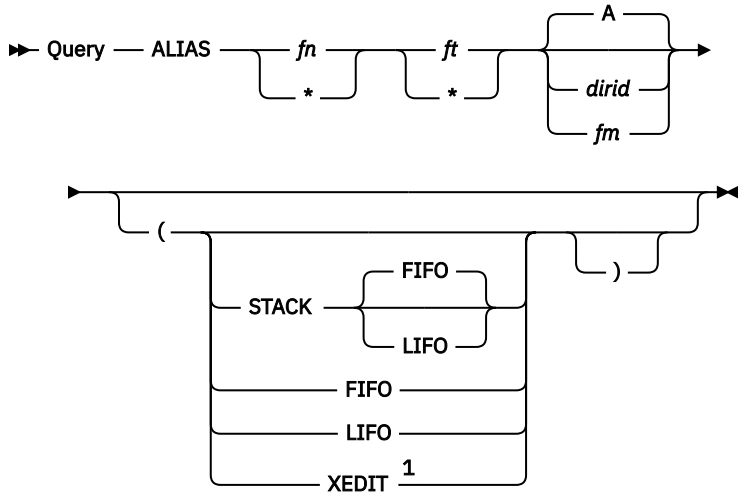
Options

For more information on the Options, Usage Notes and Error Messages that apply to all operands of the QUERY command, see "QUERY" on page 620.

Usage Notes

1. For this command, you are an accessor if you:
   - Entered a CMS ACCESS command for the directory and have not yet released it.
   - Used the CSL Open Directory for Files (DMSOPDIR) routine for the directory and have not yet issued the Close Directory (DMSCLDIR) for it.
2. The QUERY ACCESSORS command will fail if it is issued from a work unit that is active.
3. The QUERY ACCESSORS command will fail if you specify a file control directory.
4. Error messages and the heading are suppressed if QUERY ACCESSORS is invoked:
   - As a function from a program
   - From a CMS exec file that has the &CONTROL NOMSG option in effect
   - From an EXEC 2 exec where CMDCALL is not in effect
   - From a REXX exec with ADDRESS COMMAND in effect
5. A different version of this command is available for users with file pool administration authority. For more information, see the *z/VM: CMS File Pool Planning, Administration, and Operation*.

QUERY ALIAS



Notes:

[1] The XEDIT option is valid only from XEDIT.

Authorization

General User

Purpose

Use the QUERY ALIAS command to display alias information for a base file or an alias in a Shared File System (SFS) directory.

Operands

fn
is the file name of the file queried.

ft
is the file type of the file queried.

Note: You must have read or write authority to the file you are querying. Special characters (* or %) can be used to designate a set of files, providing you have appropriate authority for the directory specified by *dirid*. For FILECONTROL directories, you need read or write authority on the directory. For DIRCONTROL directories, you need DIRREAD or DIRWRITE authority on the directory.

For more information on *dirid* see "Naming Shared File System (SFS) Directories" on page 6. For more information on using special characters to do pattern matching, see "Using Pattern Matching to Specify Sets of Files" on page 14.

If you specify a set of files with a special character, all aliases or base files (depending on what you are querying) matching the specified criteria will be listed. Files you do not have read or write authority on are not listed.

dirid
is the directory queried. If *dirid* is not specified, the directory queried is the one accessed as A. You can also use a file mode for the *dirid* if the directory is already accessed. For more information on *dirid*, see "Naming Shared File System (SFS) Directories" on page 6.

Responses

```
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 only generated if output is displayed at the terminal.

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)

A*
alias pointing to a base file that is in DFSMS/VM migrated status

B*
base file that is in DFSMS/VM migrated status

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, `userid` is the owner of the base file.

nnn
is the number of aliases to the base file 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, 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, this is the file type of the base file.

dirname
is the name of the directory containing *fn2 ft2*.

Note: If no aliases are in effect, and the STACK, LIFO, or FIFO option was specified, the return code is set to 6, indicating no data was stacked.

Examples

- Querying a Base File

If you are the owner, a listing is returned of the users that have aliases to that base file, along with the number of aliases they have. If you own or have read or write authority on the directory containing the alias, the alias name is also returned.

If you are not the owner, a listing of your aliases to that base file is returned.

For example, Smith queries for aliases on the base file CODING STANDRDS in his .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      STNDARDS .
```

This response tells SMITH that four aliases exist for his base file CODING STANDRDS. The first three columns identify the queried file, column T has a ‘B’ indicating 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 the person querying does not have authority on the directory that contains the aliases.

- Querying an Alias

  The owner of the base file is returned, even if the alias has been revoked or erased. If you own or have read or write authority on the directory containing the base file the base file name is also returned. This is true even for a revoked alias.

  For example, Smith queries aliases with a file name of 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  JOHNH      1 MACRO    INFORMTN .
  MACRO    OLDSTNRD A0 E  JOHNH      1
  MACRO    PROPOSED A0 R  JOHNH      1
  MACRO    PREVIOUS A0 A* JOHNH      1
  ```

  This tells SMITH he has four aliases in his .INFO directory with a file name of MACRO. The first three columns give the full file ID of the alias found. Column T gives the status of the base file which the alias refers to. In this example, MACRO STANDRDS is an alias to base file MACRO INFORMTN in JOHNH's top directory. Alias MACRO OLDSTNRD has an ‘E’ in column T, which indicates the base file has been erased. Alias MACRO PROPOSED has an ‘R’ in column T, indicating SMITH's authority to the base file has been revoked. Alias MACRO PREVIOUS has an ‘A*’ in column T, which indicates this alias is pointing to a base file that has been placed in DFSMS/VM migrated status. In the last three entries the last three columns are blank, indicating SMITH does not have authority to JOHNH's directory, which contains three of the base files.

For more examples on using the QUERY ALIAS command, see *z/VM: CMS User's Guide*.

Response Messages

- If no alias exists for the file or directory, the response is:

  ```
  No alias exists for fn ft dirid
  ```

  or

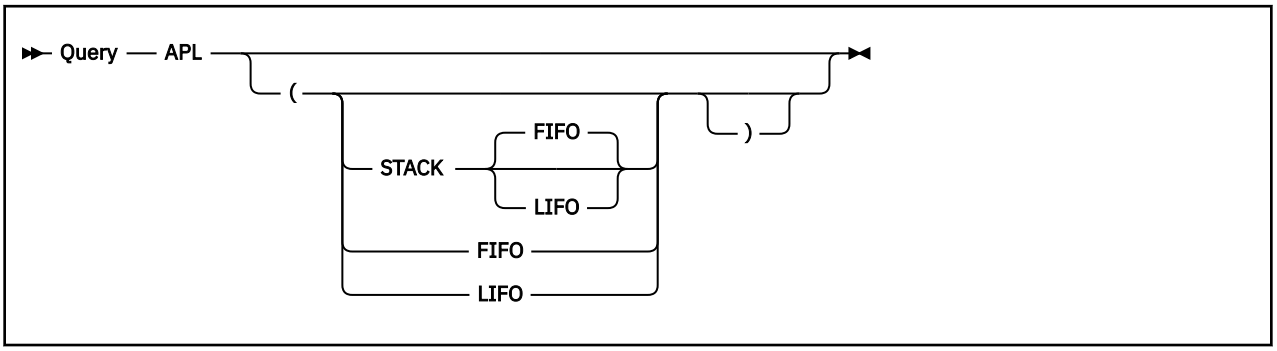
  ```
  No alias exists for dirid
  ```

Options

When using the XEDIT option, the edited file must either be variable length format or fixed format with a record length of 190.

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see “QUERY” on page 620.

QUERY APL



Authorization

General User

Purpose

Use the QUERY APL command to display the status of APL character code conversion. Use the SET APL command to activate or deactivate character code conversion.

Responses

```
APL       ON
```

or

```
APL       OFF
```

Where:

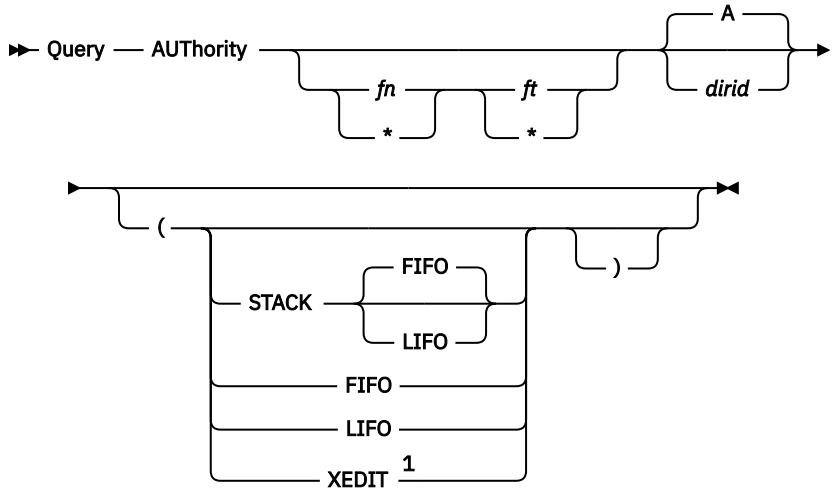
ON
indicates APL character conversion is in effect.

OFF
indicates APL character conversion is not in effect.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see “QUERY” on page 620.

QUERY AUTHORITY



Notes:

[1] The XEDIT option is valid only from XEDIT.

Authorization

General User

Purpose

Use the QUERY AUTHORITY command to display the authorities for a Shared File System (SFS) directory or for a file or files in the directory.

Operands

fn
is the file name queried.

ft
is the file type queried.

If *fn ft* is not specified, the authority you have on the directory specified with *dirid* is shown. If *fn ft* is specified as asterisks, your authorities for all the files contained in the *dirid* are shown.

dirid
is the directory queried. If *dirid* is omitted, the directory queried is the one accessed as A. You can also use a file mode for the *dirid* if the directory is already accessed. For more information on *dirid*, see “Naming Shared File System (SFS) Directories” on page 6.

Responses

- 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*

  If the directory has the file control (FILECONTROL) attribute, the following is displayed:

```
Directory = filepool:userid.n1.n2...n8
Grantee   R  W  NR  NW
userid    r  w  nr  nw
  .       .  .  .   .
  .       .  .  .   .
```

  If the directory has the directory control (DIRCONTROL) attribute, the following is displayed:

```
Directory = filepool:userid.n1.n2...n8
Grantee   R  W  DR  DW
userid    r  w  dr  dw
  .       .  .  .   .
  .       .  .  .   .
```

Where:

dr
indicates the DIRREAD authority for the listed directory.

X
means you have the authority.

–
means you do not have the authority or the authority is revoked.

P
means the object is protected by an External Security Manager (ESM). Any authorizations displayed for this object are SFS authorizations. ESM authorizations, which are not displayed, will override SFS authorizations. Use commands or functions provided by your ESM to determine your ESM authorization.

dw
indicates the DIRWRITE authority for the listed directory.

X
means you have the authority.

–
means you do not have the authority or the authority is revoked.

P
means the object is protected by an External Security Manager (ESM). Any authorizations displayed for this object are SFS authorizations. ESM authorizations, which are not displayed, will override SFS authorizations. Use commands or functions provided by your ESM to determine your ESM authorization.

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, this column contains a dash. If you have queried a directory, this field is omitted.

nr
indicates the NEWREAD authority for the listed directory.

X
means you have the authority.

–
means you do not have the authority or the authority is revoked.

P
means the object is protected by an External Security Manager (ESM). Any authorizations displayed for this object are SFS authorizations. ESM authorizations, which are not displayed, will override SFS authorizations. Use commands or functions provided by your ESM to determine your ESM authorization.

nw
indicates the NEWWRITE authority for the listed directory.

X
means you have the authority.

–
means you do not have the authority or the authority is revoked.

P
means the object is protected by an External Security Manager (ESM). Any authorizations displayed for this object are SFS authorizations. ESM authorizations, which are not displayed, will override SFS authorizations. Use commands or functions provided by your ESM to determine your ESM authorization.

type
is the type of file or directory as follows:

ALIAS
alias for a base file

BASE
base file

DIR
directory having the file control (FILECONTROL) attribute

DIRC
directory having the directory control (DIRCONTROL) attribute

ERASED
erased (the base file has been erased)

REVOKED
revoked (authority to the base file has been revoked).

ALIAS*
alias pointing to a base file which has been placed in DFSMS/VM migrated status

BASE*
base file which has been placed in DFSMS/VM migrated status

EXTRNL
external object

userid
is the user ID who owns or has been granted authority on the file or directory or external object. This will be <PUBLIC> if authority was granted to all users who can connect to the file pool. For external objects, it is the user ID who owns the directory containing the external object.

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 the authority is revoked. For external objects, a dash is always displayed.

P
means the object is protected by an External Security Manager (ESM). Any authorizations displayed for this object are SFS authorizations. ESM authorizations, which are not displayed, will override SFS authorizations. Use commands or functions provided by your ESM to determine your ESM authorization.

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 the authority is revoked. For external objects, a dash is always displayed.

P
means the object is protected by an External Security Manager (ESM). Any authorizations displayed for this object are SFS authorizations. ESM authorizations, which are not displayed, will override SFS authorizations. Use commands or functions provided by your ESM to determine your ESM authorization.

Examples

John wants to see the authorities he has been granted on another user's directory, so he issues,

```
query aut * * smith.info
```

and this information is displayed, assuming John has read or write authority to the FILEPOOL:SMITH.INFO directory:

```
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     -  -
TEST                  DIRC    JOHN     X  X
```

John has read/write authority on the file EXAMPLE SCRIPT. He has read authority on the FILECONTROL directory SMITH.INFO.PROJ2. There is an alias, CODING STANDRDS, he was granted read authority on, but the authority is ESM-protected. John's authority to use the alias, MACRO PROPOSED, has been revoked. John also has DIRWRITE authority on the SMITH.INFO.TEST directory. When DIRCONTROL directories are displayed in this situation, an ‘X’ in the R column implies DIRREAD authority. An ‘X’ in both the R and W columns implies DIRWRITE authority.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see “QUERY” on page 620.

Usage Notes

1. When using the XEDIT option, the edited file must either be variable length format or fixed format with a record length of 165.
2. If you own the file, directory, or external object, a list of user IDs, including your own, is returned with the authority granted to the file, directory, or external object. If you are not the owner, only your authority on the object is returned. For external objects, your authority is always the authority you have on its directory.

3. When *fn ft* is specified and the parent directory has the file control attribute, you must have read or write authority for the specified file. If the parent directory has the directory control attribute, you need directory control read (DIRREAD) authority or directory control write (DIRWRITE) authority on the parent directory.
4. If the file is an alias, the alias is displayed with the authority you have on the base file.
5. Special characters (* or %) can be used to designate a set of files, providing you have appropriate authority for the directory specified by *dirid*. For FILECONTROL directories, you need read or write authority on the directory. For DIRCONTROL directories, you need DIRREAD or DIRWRITE authority on the directory.

   For more information on *dirid* see "Naming Shared File System (SFS) Directories" on page 6. For more information on using special characters to do pattern matching, see "Using Pattern Matching to Specify Sets of Files" on page 14.

   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,

   ```
   PHILLIPS NOTEBOOK
   PHILLIPSNOTES
   ```

   where PHILLIPS NOTEBOOK is a file and PHILLIPSNOTES is a subdirectory. Issuing,

   ```
   query authorit phil* n* a
   ```

   would find both PHILLIPS NOTEBOOK and PHILLIPSNOTES because PHILLIPSNOTES contains more than eight characters and is matched as if it had a file name of PHILLIPS and a file type of NOTES. Issuing,

   ```
   query authorit phillipsnotes * a
   ```

   would also find both PHILLIPS NOTEBOOK and PHILLIPSNOTES because only the first eight characters are used as the file name. Therefore, any file or alias with the name of PHILLIPS, or any subdirectory with PHILLIPS as the first eight characters in its name would be listed.
6. When you specify only a *dirid*, additional columns indicate DIRREAD, DIRWRITE, NEWREAD, and NEWWRITE authority for the directory. A different set of columns is displayed for DIRCONTROL directories and for FILECONTROL directories.

   For DIRCONTROL directories, two columns (named DR and DW) indicate whether there is directory control read (DIRREAD) or directory control write (DIRWRITE) authority. For more information on these authorities, see "GRANT AUTHORITY" on page 375. When the DR or DW column contains an X, the corresponding R or W column will also contain an X because these authorities are derived from the directory control authority.

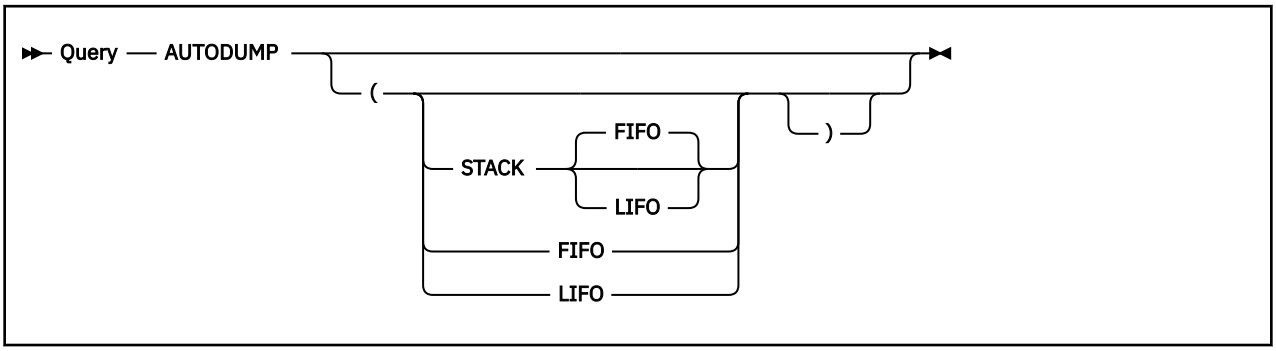
   Even though an X is shown, you do not really have READ or WRITE authority, but only DIRREAD or DIRWRITE.

   For FILECONTROL directories two columns (named NR and NW) indicate whether there is NEWREAD and NEWWRITE authority. For more information about these authorities, see "GRANT AUTHORITY" on page 375. The R and W columns are independent of the NR and NW columns. That is read and write authority can be granted independent of NEWREAD and NEWWRITE authority.

   Note: The information for the directory is displayed if you lack read and write authority but have either NEWREAD or NEWWRITE authority.
7. When *fn ft* is specified, the display shows read (R) or write (W) authority for the indicated file(s). For DIRCONTROL directories, these file authorities are derived from the DIRREAD and DIRWRITE levels of authority. Individual file authorizations cannot be granted for files that reside in DIRCONTROL directories.
8. QUERY AUTHORITY returns the latest valid information, even for accessed DIRCONTROL directories. Unlike file data, your view of authority information for DIRCONTROL directories is not frozen when you access the directory read-only.

9. If you specify pattern-matching characters for *fn ft*, directory names may be displayed as well. For FILECONTROL directories, an 'X' in the R or W columns indicates read or write authority to the directory. You cannot tell from the display whether you also have NEWREAD or NEWWRITE authority on the directory. For DIRCONTROL directories, an 'X' in the R or W columns indicates DIRREAD or DIRWRITE authority.
10. External objects have authority of their own. The remote name an external object contains may be queried (with the DMSQOBJ CSL routine) by anyone with read authority to the parent directory.
11. If <PUBLIC> is shown when you enter QUERY AUTHORITY, and public authority does not appear to be in effect, it is possible authority was granted to the user IDs of * or <PUBLIC>. Starting with z/VM version 4 release 4, the GRANT and REVOKE AUTHORITY commands issue an error message indicating "user ID is not valid" for the user IDs of * or <PUBLIC>.

QUERY AUTODUMP



Authorization

General User

Purpose

Use the QUERY AUTODUMP command to display the current setting of the SET AUTODUMP command. Use the SET AUTODUMP command to control dump creation by CMS. For more information, see "SET AUTODUMP" on page 907.

Responses

```
AUTODUMP =  OFF
```

or

```
AUTODUMP =  CMS
```

or

```
AUTODUMP =  CMS  ENTIREVM
```

or

```
AUTODUMP =  ALL
```

or

```
AUTODUMP =  ALL  ENTIREVM
```

Where:

OFF
indicates no dump will be taken. This is the initial setting.

CMS
indicates a dump will be taken whenever an irrecoverable CMS system abend occurs and CMS enters a disabled wait state.

ALL
indicates a dump will be taken for all abends within the virtual machine, both recoverable and irrecoverable.

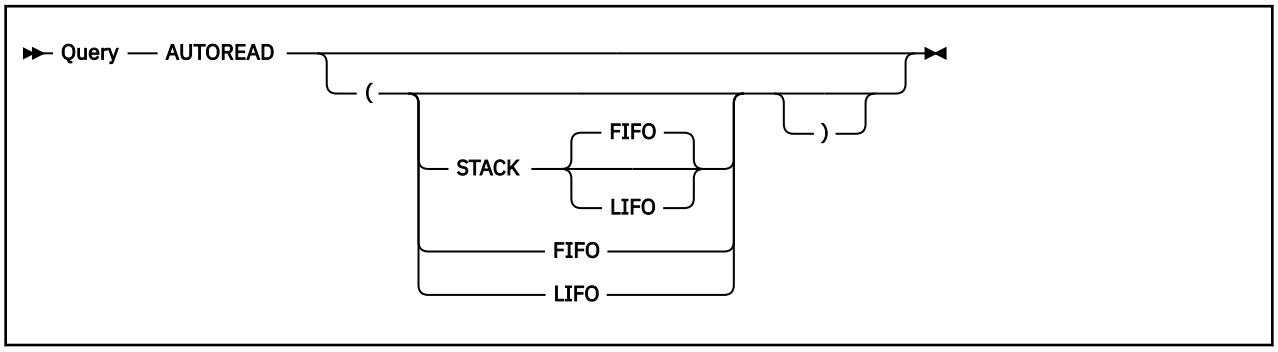
ENTIREVM
when returned with CMS or ALL, indicates a dump will be taken of the entire virtual machine, along with any DCSSs currently in use and any data spaces currently in use which contain SFS data. If not

returned, the default dump will include DMSNUC, the loader tables, the storage management work area and the page allocation table.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see “QUERY” on page 620.

QUERY AUTOREAD



Authorization

General User

Purpose

Use the QUERY AUTOREAD command to display the status of the console read. Use the SET AUTOREAD command to change the AUTOREAD setting. For more information, see “SET AUTOREAD” on page 910.

Responses

```
AUTOREAD =    ON
```

or

```
AUTOREAD =    OFF
```

Where:

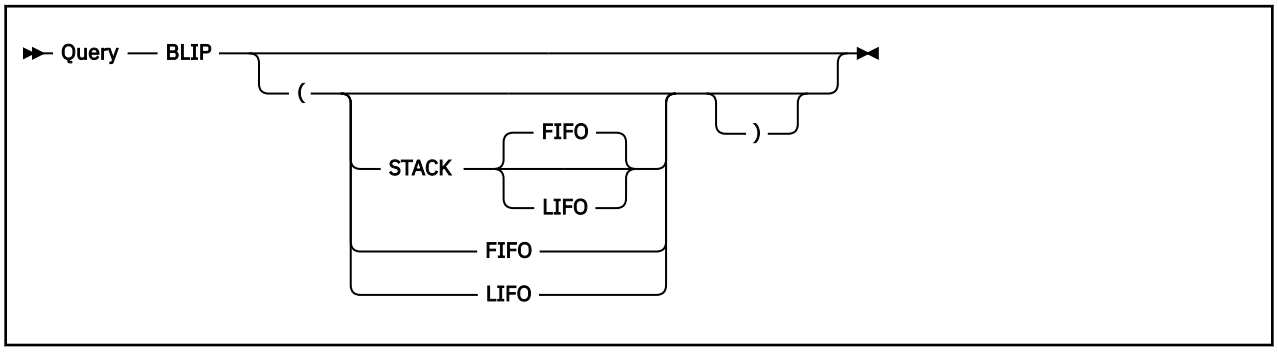
ON
indicates a console read is issued immediately after command execution.

OFF
indicates no console read is issued until the enter key (or its equivalent) is pressed.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see “QUERY” on page 620.

QUERY BLIP



Authorization

General User

Purpose

Use the QUERY BLIP command to maintain for compatibility only. CMS does not support the BLIP facility.

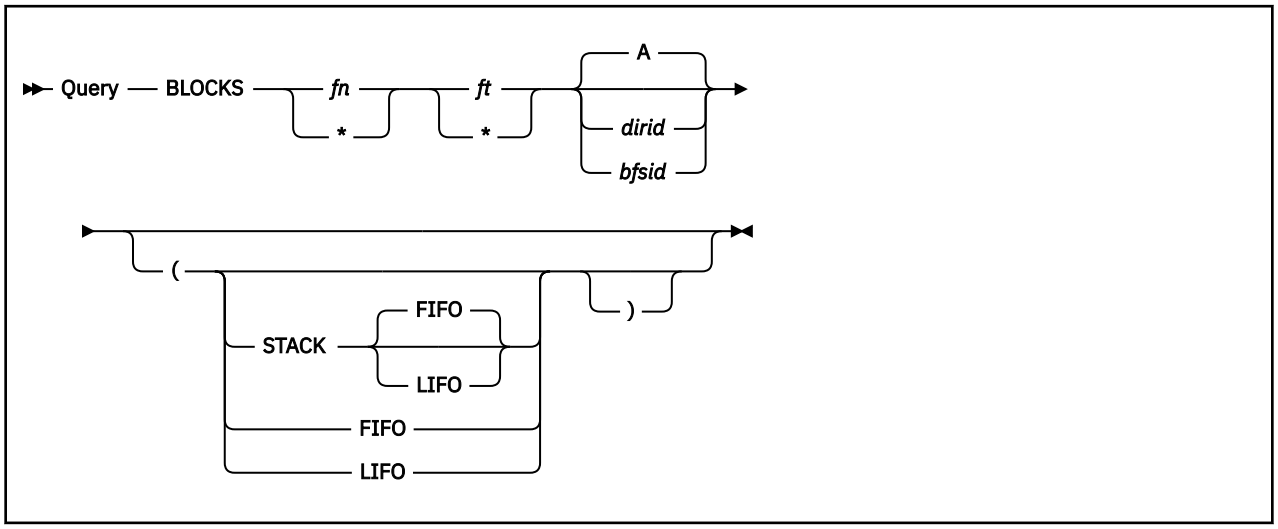
Responses

```
BLIP =   OFF
```

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see “QUERY” on page 620.

QUERY BLOCKS



Authorization

General User for SFS files. An SFS administrator may enter this command for byte file system (BFS) files.

Purpose

Use the QUERY BLOCKS command to determine how much space an SFS file, BFS regular file, or set of files is using for its data and how much space the system is using for the file(s).

Operands

fn
is the file name queried.

ft
is the file type queried.

If *fn ft* is specified as asterisks, the amount of space used by all the files contained in the *dirid* is shown.

dirid
is the directory queried. If *dirid* is omitted, the directory queried is the one accessed as A. You can also use a file mode for the *dirid* if the directory is already accessed. For more information on *dirid*, see “Naming Shared File System (SFS) Directories” on page 6.

bfsid
identifies the byte file system (BFS).

For a BFS file, *fn* and *ft* are system-generated values that together uniquely identify the file within the BFS.

Responses

- For

```
Directory = dirname
Filename Filetype Fm Type Datablocks Systemblocks
fn       ft       fm type dblocks    sblocks
fn       ft       fm type dblocks    sblocks
 .        .        .  .    .          .
```

```
.        .        . .    .          .
.        .        . .    .          .
```

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:

Filename
is the name of the file or subdirectory.

Filetype
is the file type.

Fm
is the file mode of the directory in which the file or subdirectory resides. If the directory is not accessed, this column contains a dash (–). For example:

```
query blocks * * fpool:mary.
```

returns:

```
Directory = FPOOL:MARY.
Filename Filetype Fm Type    Datablocks  Systemblocks
EXAMPLE  SCRIPT   A1 BASE    1           0
CODING   STANDRDS A1 ALIAS   2           1
SAMPLES           B  DIR     -           -
TESTCASE          B  DIRC    -           -
CODING3  STANDRDS A6 ERASED  -           -
CODING4  STANDRDS A1 REVOKED -           -
OTHER    STANDRDS A1 EXTRNL  -           1
```

type
specifies one of these:

- *BASE* specifies a base file in a directory
- *DIR* specifies a FILECONTROL directory
- *DIRC* specifies a DIRCONTROL directory
- *ALIAS* specifies an alias in a directory
- *ERASED* specifies an erased alias
- *REVOKED* specifies a revoked alias
- *ALIAS** specifies an alias of a file in DFSMS/VM migrated status
- *BASE** specifies a base file in DFSMS/VM migrated status
- *EXTRNL* specifies an external object.

datablocks
is the actual number of blocks used by the data in the file only. In general, this will be the same as the *noblks* value returned by FILELIST and LISTFILE. However, for sparse variable files, the empty blocks which are not allocated are not included in the count. The FILELIST and LISTFILE commands do include empty blocks for the variable files however, they do not include them for fixed files.

systemblocks
is the additional number of blocks used by the system for the file, in addition to the data blocks.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see “QUERY” on page 620.

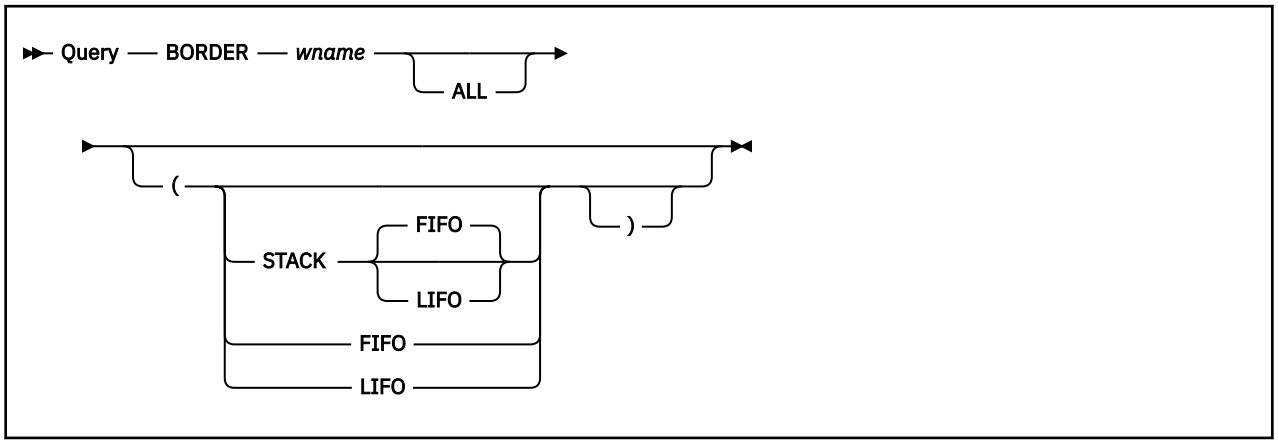
Usage Notes

1. Only file and system blocks are counted. Space used by the SFS catalogs is not reflected here.
2. An external object itself does not use file space. However, it can have system blocks associated with it.

3. If you specify *fn ft*, you need read authority on the directory containing that file.
4. All blocks for which counts are returned are 4K bytes in length.
5. A value for *datablocks* is returned for base files and active aliases (that is, not aliases not revoked nor erased) only. The value displayed for an alias actually refers to the base file. QUERY BLOCKS displays a dash (–) in the *datablocks* column for all other types of file pool objects.
6. A value for *systemblocks* is returned for base files, active aliases and external objects only. The value displayed for an alias actually refers to the base file. QUERY BLOCKS displays a dash (–) in the *systemblocks* column for all other types of file pool objects.

   Byte file system (BFS) files larger than one block in size will contain a non-zero value in the system block field. When these files are read using DMSOPEN/DMSREAD, DMSOPBLK/DMSRDBLK, or DMSOPDBK/DMSRDDBK, system blocks are created. The system blocks field of the QUERY BLOCKS output will contain the number of system blocks created when these routines are used to read a BFS file.

QUERY BORDER



Authorization

General User

Purpose

Use the QUERY BORDER command to display whether the window borders are set to ON the edge names, characters, and border attributes. Use the SET BORDER command to define borders around windows. For more information, see "SET BORDER" on page 912.

Operands

wname
is the name of the window whose border information is to be displayed.

ALL
specifies the border attributes are to be displayed, as well as the same information as for QUERY *wname*.

Responses

- For QUERY BORDER *wname*

```
BORDER wname {ON|OFF} ([TOP char] [BOTTOM char]
[LEFT char] [RIGHT char]
```

Where:

wname
is the name of the window.

char
is the character assigned to the border edge.

- For QUERY BORDER *wname* ALL

```
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 HIGH or NOHIGH.

color
is the color of the border.

exthi
is the extended highlighting of the border.

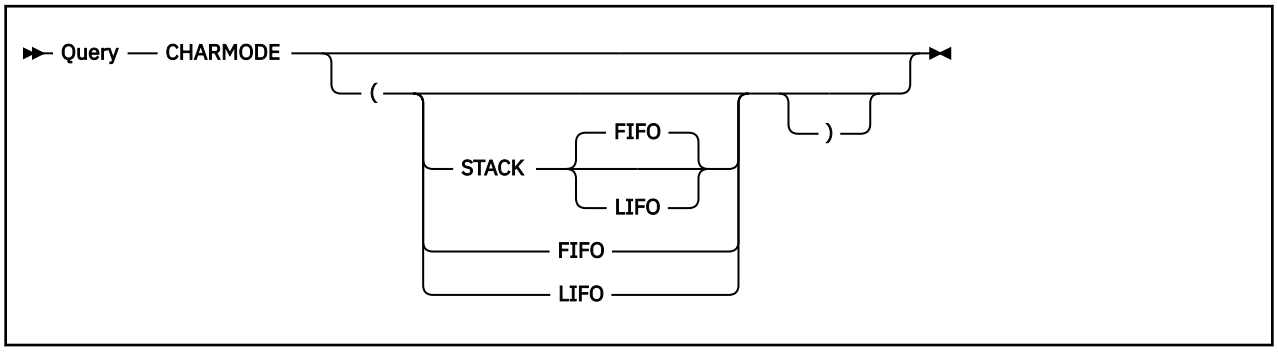
psset
is the Programmed Symbol Set of the border.

Note: When using the ALL operand, the response may be too long for one line. If so, it wraps to the next one.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see "QUERY" on page 620.

QUERY CHARMODE



Authorization

General User

Purpose

Use the QUERY CHARMODE command to display whether character or field attributes are used when displaying virtual screen data on the physical screen. For more information on how to to change the CHARMODE setting, see “SET CHARMODE” on page 915.

Responses

```
CHARMODE      ON
```

or

```
CHARMODE      OFF
```

Where:

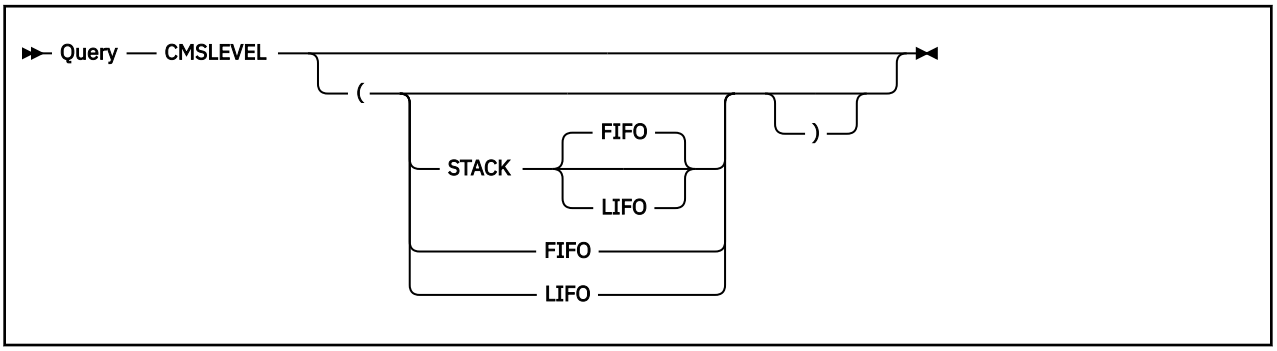
ON
indicates character attributes are used when displaying virtual screen data on the physical screen.

OFF
indicates field attributes are used when displaying virtual screen data on the physical screen.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see “QUERY” on page 620.

QUERY CMSLEVEL



Authorization

General User

Purpose

Use the QUERY CMSLEVEL command to display the release (level) and service level of CMS. (You can use the DMSQEFL CSL routine to return this information to a program.)

Responses

The format of the response is:

```
CMS Level nn, Service Level yyxx
```

CMS Level *nn*
is the CMS level number for the CMS level you are running.

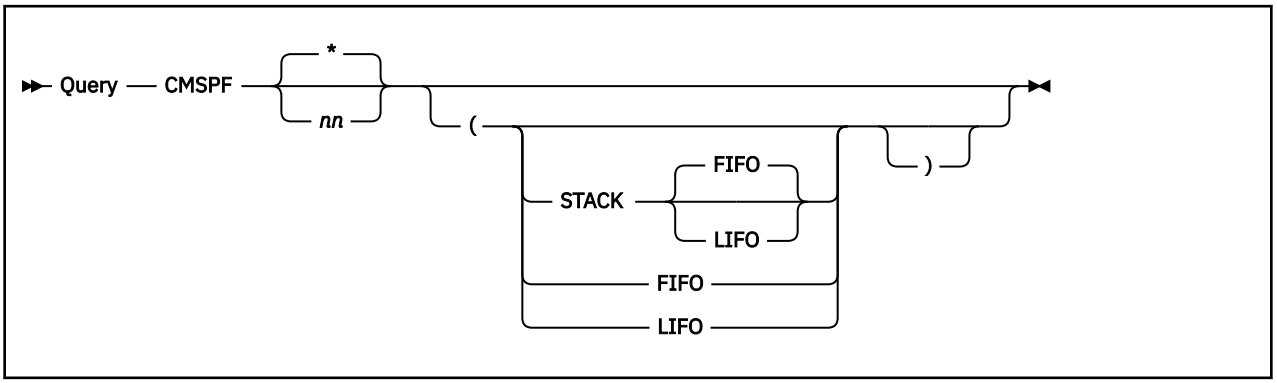
Service Level *yyxx*
is the software service level number. The number indicates the most recent RSU service tape that has been applied. *yy* is the last two digits of the year. *xx* is the sequential number of the RSU tape for that year. It cannot indicate which individual updates have been incorporated into CMS. The system programmer can find out what individual updates have been incorporated by using the SERVICE command. For more information, see the *z/VM: Service Guide*.

If z/Architecture CMS (z/CMS) is running in the virtual machine, the response indicates z/CMS instead of CMS.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see "QUERY" on page 620.

QUERY CMSPF



Authorization

General User

Purpose

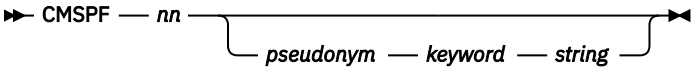
Use the QUERY CMSPF command to display the definition of a specific CMS PF key. Use the SET CMSPF command to set CMS PF keys. For more information, see "SET CMSPF" on page 917.

Operands

nn
specifies the specific CMS PF key queried.

***
specifies all full-screen CMS PF keys are to be queried. This is the default.

Responses



Where:

nn
is the number of the PF key.

pseudonym
is a nine-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. This string will begin in column 32 of the CMSPF Key definition.

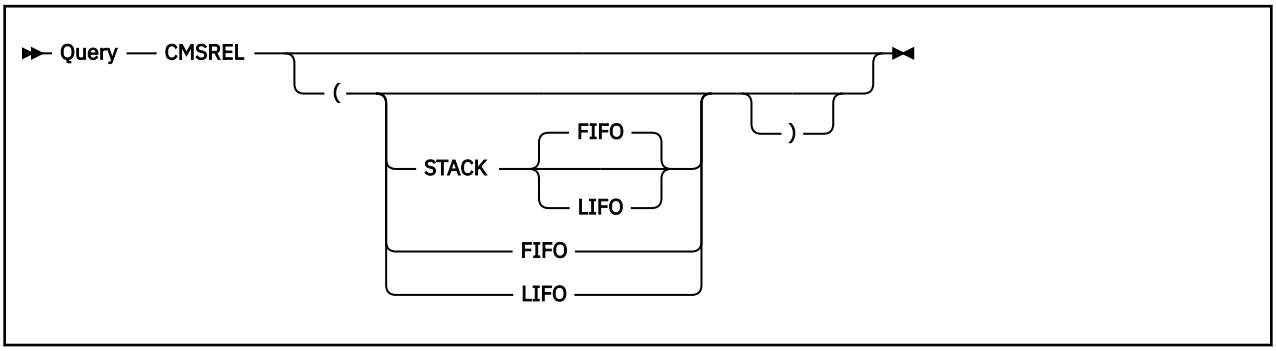
A CMSPF key that is undefined is displayed as:

```
CMSPF nn
```

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see “QUERY” on page 620.

QUERY CMSREL



Authorization

General User

Purpose

Use the QUERY CMSREL command to return the licensed program name and release level. You may also use the DMSQEFL CSL routine to return this information to a program.

Responses

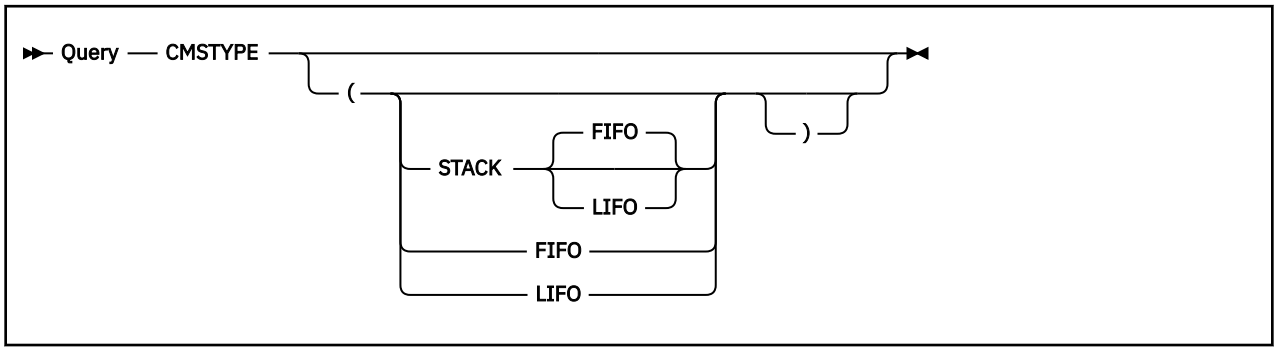
The format of the response is:

```
z/VM Version n Release n.n
```

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see “QUERY” on page 620.

QUERY CMSTYPE



Authorization

General User

Purpose

Use the QUERY CMSTYPE command to indicate the status of the CMS terminal display. This command should only be used from an exec environment and only with the STACK or LIFO/FIFO options, or both, specified. Use the SET CMSTYPE command to change the CMSTYPE setting. For more information, see “SET CMSTYPE” on page 919.

Responses

```
CMSTYPE = HT
```

or

```
CMSTYPE = RT
```

Where:

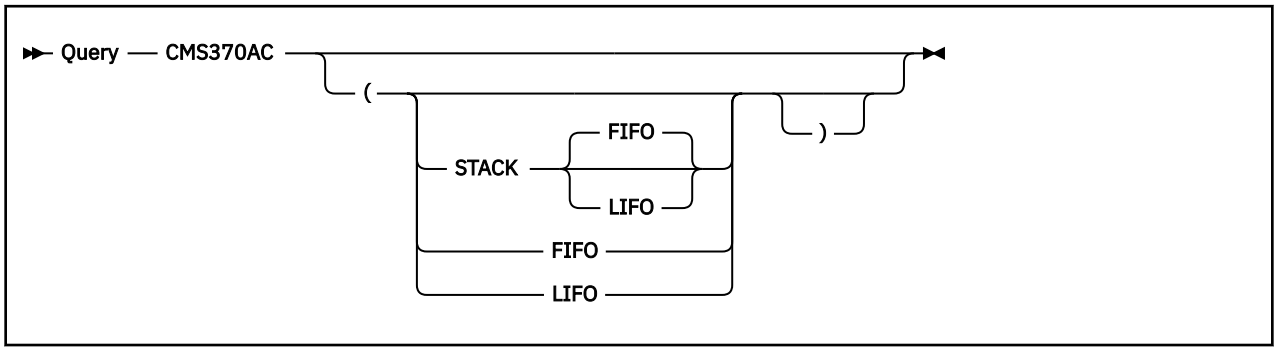
HT
indicates the CMS terminal display within an exec is suppressed. All CMS terminal display from an exec, except for CMS error messages with a suffix letter of ‘S’ or ‘T’, is suppressed until the end of the exec file or until the SET CMSTYPE RT command is executed.

RT
indicates the CMS terminal display is not suppressed.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see “QUERY” on page 620.

QUERY CMS370AC



Authorization

General User

Purpose

Use the QUERY CMS370AC command to display whether the SET CMS370AC setting is ON or OFF.

Responses

```
CMS370AC = ON
```

or

```
CMS370AC = OFF
```

Where:

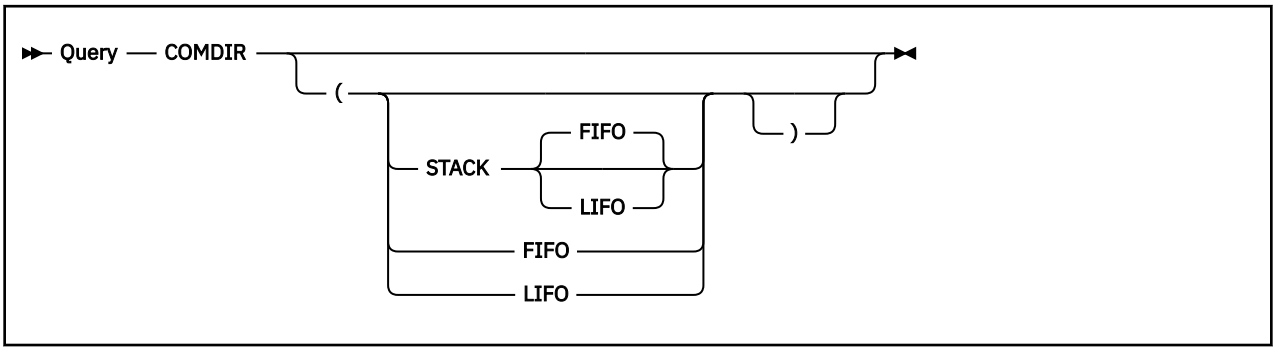
ON
Indicates the CMS370AC facility is ON, which allows support for System/370 applications that will not execute properly in XA and XC virtual machines.

OFF
Indicates the CMS370AC facility is OFF.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see “QUERY” on page 620.

QUERY COMDIR



Authorization

General User

Purpose

Use the QUERY COMDIR command to display the current CMS communications directory settings. Use the SET COMDIR command, to change these settings. For more information, see "SET COMDIR" on page 921.

Responses

```
SYSTEM  status  loadstatus  fileid
USER    status  loadstatus  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
indicates name resolution is on.

OFF
indicates name resolution is off.

loadstatus
indicates whether the communications directory file is currently loaded.

The *loadstatus* is either:

STATIC
indicates the file was loaded into storage when the last SET COMDIR FILE or SET COMDIR RELOAD was entered.

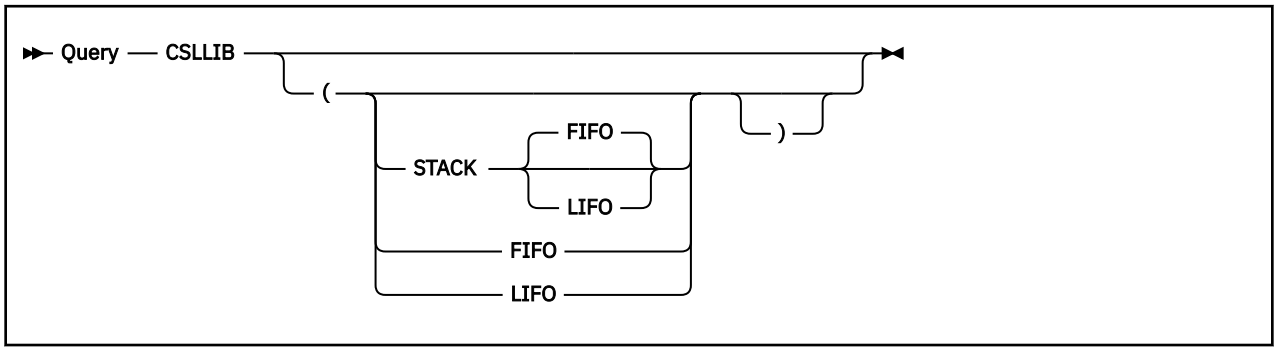
UNLOADED
indicates the communications directory is not loaded.

fileid
is the name of the directory. A response of "(none)" indicates no directory is loaded.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see “QUERY” on page 620.

QUERY CSLLIB



Authorization

General User

Purpose

Use the QUERY CSLLIB command to display the names of the callable service libraries (with a file type of CSLLIB or CSLSEG) in the current library search order. (This means all callable service libraries specified on the last GLOBAL CSLLIB command, if any.)

Responses

```
CSLLIB = libname1 . . . libname8
  .          .              .
  .          .              .
  .          .              .
```

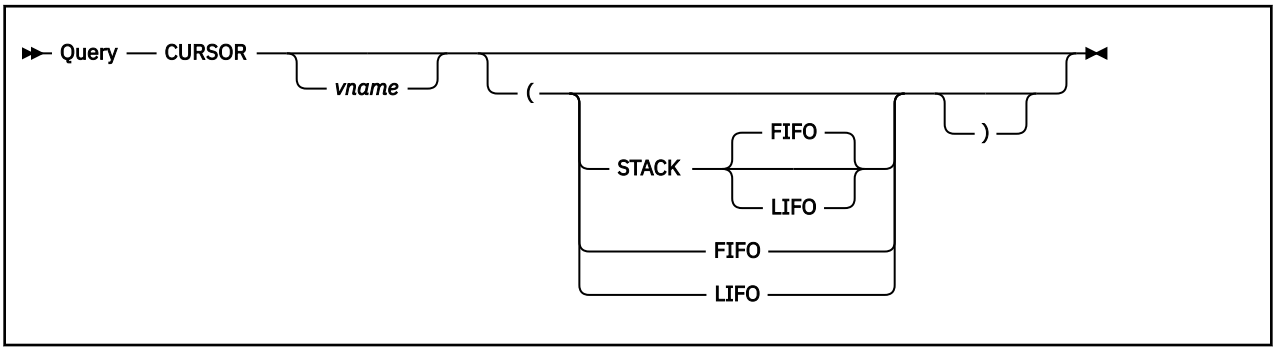
Up to eight names are displayed per line, for as many lines as necessary. If no CSLLIBs are found in the search order, the following message is displayed at the terminal:

```
CSLLIB = NONE
```

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see “QUERY” on page 620.

QUERY CURSOR



Authorization

General User

Purpose

Use the QUERY CURSOR command to display the location of the cursor on the physical screen or a virtual screen.

Operands

vname
is the name of the virtual screen in which the current cursor location is to be returned.

Responses

- For QUERY CURSOR

```
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.

vline
is the line number of the virtual screen. A value of zero (0) indicates 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.

- For QUERY CURSOR *vname*

```
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 the cursor is below the current bottom.

vcol
is the column number of the virtual screen.

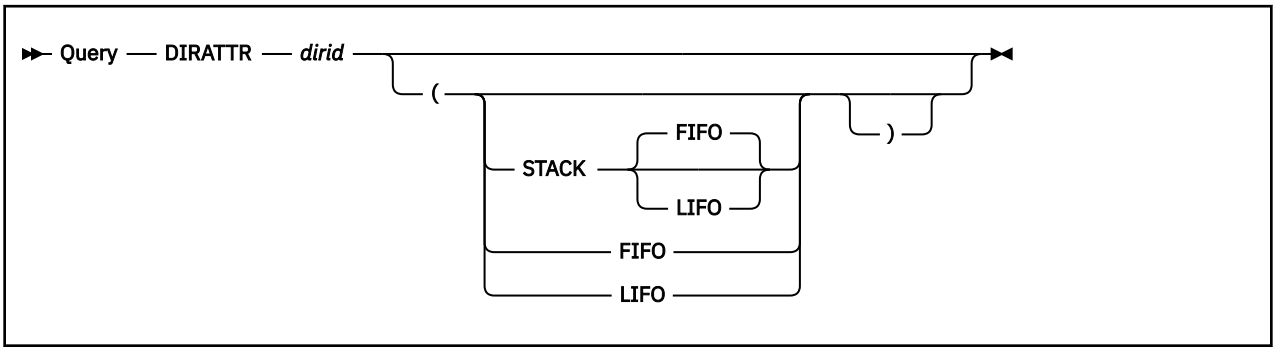
RESERVED | DATA
is the area where the cursor is set.

Note: If the cursor has not yet been set, *vline* and *vcol* are -1.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see "QUERY" on page 620.

QUERY DIRATTR



Authorization

General User

Purpose

Use the QUERY DIRATTR command to display the status of the directory control attribute.

Operands

dirid
identifies the SFS directory for which you want to display the directory control attribute. You can also use a file mode for the *dirid* if the directory is already accessed. For a more information on *dirid*, see "Naming Shared File System (SFS) Directories" on page 6.

Responses

QUERY DIRATTR displays one of these:

- FILECONTROL
- DIRCONTROL

FILECONTROL and DIRCONTROL are mutually exclusive attributes of SFS directories. For more about the attributes, see "CREATE DIRECTORY" on page 97.

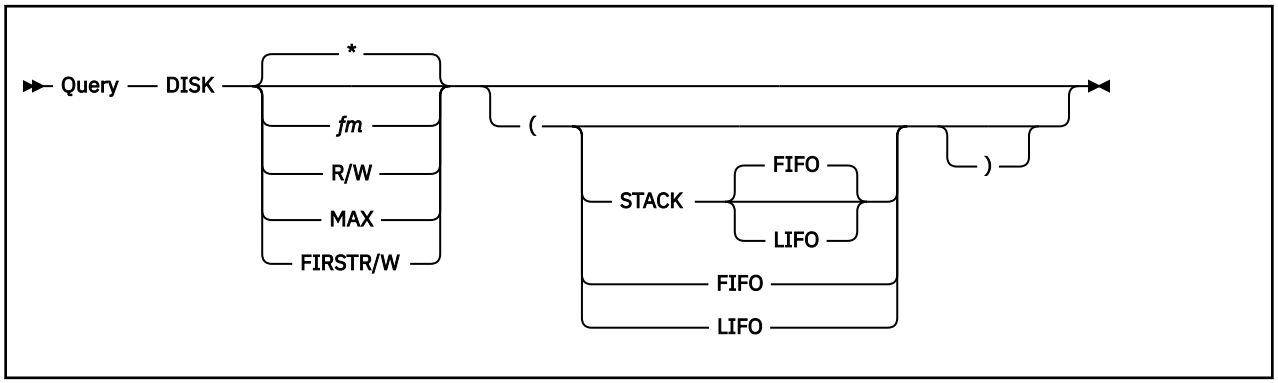
Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see "QUERY" on page 620.

Usage Notes

1. The QUERY DIRATTR command fails if it is issued from a work unit that is active.
2. You can display the directory control attributes for directories you own. You can also display it for other user's directories if you have at least read authority (for FILECONTROL directories) or DIRREAD authority (for DIRCONTROL directories).

QUERY DISK



Authorization

General User

Purpose

Use the QUERY DISK command to display the status of a disk or Shared File System (SFS) directory.

Operands

fm
specifies the file mode letter of the directory queried.

***
specifies all accessed CMS disks and SFS directories are to be queried. This is the default.

R/W
queries the status of all CMS disks and SFS directories that have been accessed in read/write mode.

MAX
queries the status of the disk or directory accessed in read/write mode that has the most available space. SFS directories are considered only when you have R/W authority to those directories that are accessed R/W.

FIRSTR/W
queries the status of the first CMS disk or SFS directory in the search order accessed in read/write mode.

Responses

One line is displayed for each accessed disk or directory.

```
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 number for a disk or *DIR* for an SFS directory. If the directory has the directory control (DIRCONTROL) attribute, *DIRC* is displayed instead of DIR.

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 or smaller than the number actually available. CMS uses the device size field in the format 4 DSCB in the OS VTOC to report this size. 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. When the calculated percentage is zero and there are files on the disk, the percentage will be displayed as **01**. 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.

Response Messages

- If the disk or directory with the specified file mode is not accessed, the response is:

```
Disk fm not accessed
```

- If there are no disks or directories accessed in read/write mode, the response to `QUERY DISK R/W` or `QUERY DISK FIRSTR/W` 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 to `QUERY DISK MAX` is:

```
No read/write disk or directory with space is accessed
```

  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.

Examples

This is an example of the results of QUERY DISK:

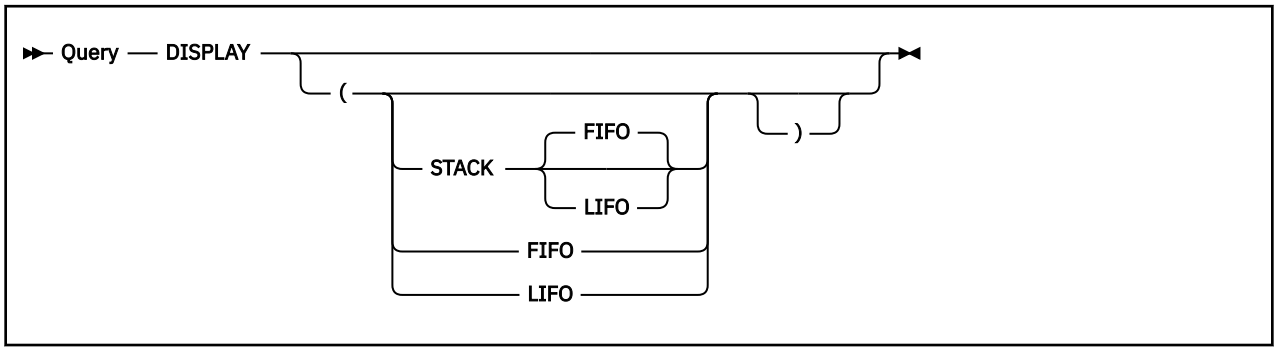
```
LABEL  VDEV M  STAT   CYL TYPE BLKSZ   FILES  BLKS USED-(%) BLKS LEFT  BLK TOTAL
SMI191 191  A   R/W    25 3380 4096     1325       2747-73       1003       3750
```

```
OSTEST 18F1 B/B R/O  3339 3390           OS
VSAM01 300  C   R/W   200 3380          DOS
-      DIR  D   R/W     -    - 4096      31           -         -         -
ESA190 190  S   R/O    86 3380 4096     350       9197-71      3703     12900
ESA19E 19E  Y/S R/O   200 3380 4096     903      14939-50     15061     30000
```

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see “QUERY” on page 620.

QUERY DISPLAY



Authorization

General User

Purpose

Use the QUERY DISPLAY command to display the characteristics of the physical screen.

Responses

```
DISPLAY lines cols devtype addrtype dbcs color exthi pss pssets outline
```

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 (Alpha Numeric Replacement) - such as 3270 to 3277 type displays.
NDS (New Display Systems) - 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 Programmed Symbol Sets (PSSET) currently loaded. 0, 1, 8, A, B, C, D, E, and F can be returned. Each character represents a PSSET. The PSSET value of 0 is always displayed.

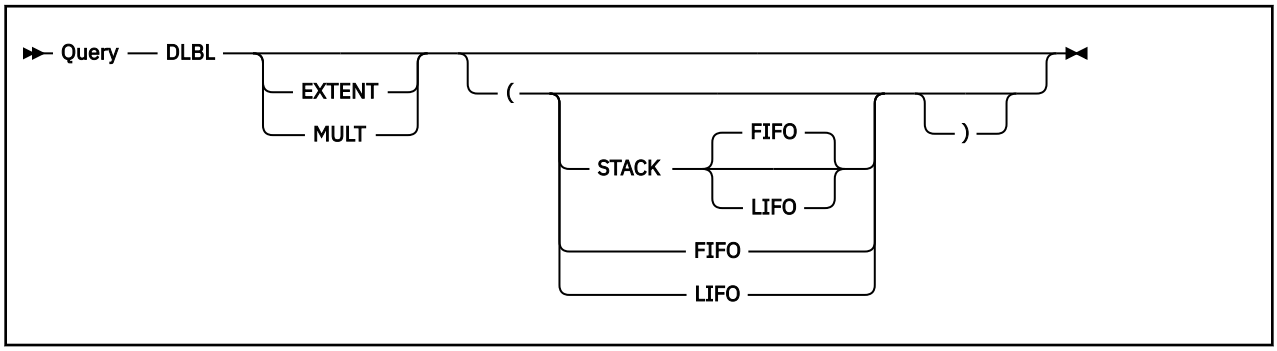
Note: The response for *pss* (PSS or NOPSS) is for loadable programmed symbol sets only. PSS 1 and 8 are nonloadable programmed symbol sets. It is therefore possible to get a *pss* response of NOPSS with pssets 0, 1, 8 or all three.

outline
specifies whether the device supports field outlining. OUTLINE or NOOUTLIN is returned.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see "QUERY" on page 620.

QUERY DLBL



Authorization

General User

Purpose

Use the QUERY DLBL command to display the contents of the current data set definitions. For more information, see "DLBL" on page 185.

Operands

EXTENT
displays the actual extents entered for a VSAM data set at DLBL definition time. It will provide the following information to the user: DDNAME, MODE, LOGUNIT, and EXTENT. For more information, see "Responses" on page 668.

MULT
displays the volumes on which all multivolume data sets reside. It will provide the following information to the user: DDNAME, MODE, and LOGUNIT. For more information, see "Responses" on page 668.

Responses

Entering the command (or simply: DLBL) yields the following information:

DDNAME
the VSE file name or OS ddname.

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 EXTENT definitions are active, the response is:

No user defined EXTENTs 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 MULT definitions are active, the response is:

No user defined MULTs 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.
```

Examples

Example with No EXTENT Definitions in Effect

If you have previously entered:

```
assgn sys001 b
dlbl junk b dsn pdsf.l80.b80 ( sys001
```

and you then enter:

```
query dlbl
```

this information is displayed:

```
DDNAME  MODE LOGUNIT  TYPE CATALOG EXT VOL BUFSPC PERM DISK DATASET.NAME
JUNK     B1  SYS001   SEQ                          NO   DOS PDSF.L80.B80
```

If you enter:

```
query dlbl extent
```

you will receive the message:

```
No user defined EXTENT in effect
```

Example with EXTENT Definitions in Effect

If you enter the following commands:

```
assgn sys001 b
dlbl junk b dsn pdsf.l80.b80 ( sys001 extent
```

you will receive the following message:

```
DMSDLB331R  Enter extent specifications:
```

If you then enter

```
15 15, 45 15
(null line)
```

in response to the prompt, and enter

```
query dlbl extent
```

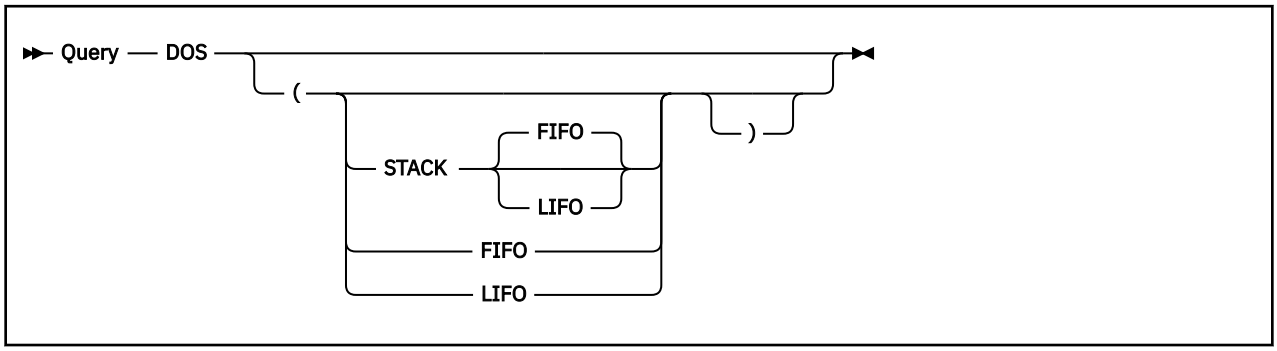
the following information is displayed:

```
DDNAME  MODE LOGUNIT            EXTENT
JUNK     B   SYS001          15         15
         B   SYS001          45         15
```

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see “QUERY” on page 620.

QUERY DOS



Authorization

General User

Purpose

Use the QUERY DOS command to display whether the CMS/DOS environment is active or not. Use the SET DOS command to change the setting. For more information, see “SET DOS” on page 923.

Responses

```
DOS =   ON
```

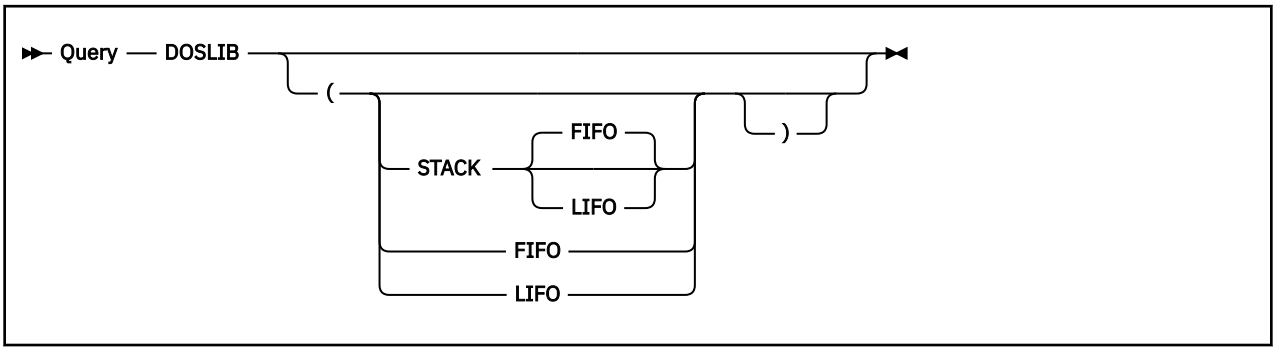
or

```
DOS =   OFF
```

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see “QUERY” on page 620.

QUERY DOSLIB



Authorization

General User

Purpose

Use the QUERY DOSLIB command to display 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 command, if any).

Responses

```
DOSLIB = libname1 … libname8
  .         .               .
  .         .               .
  .         .               .
```

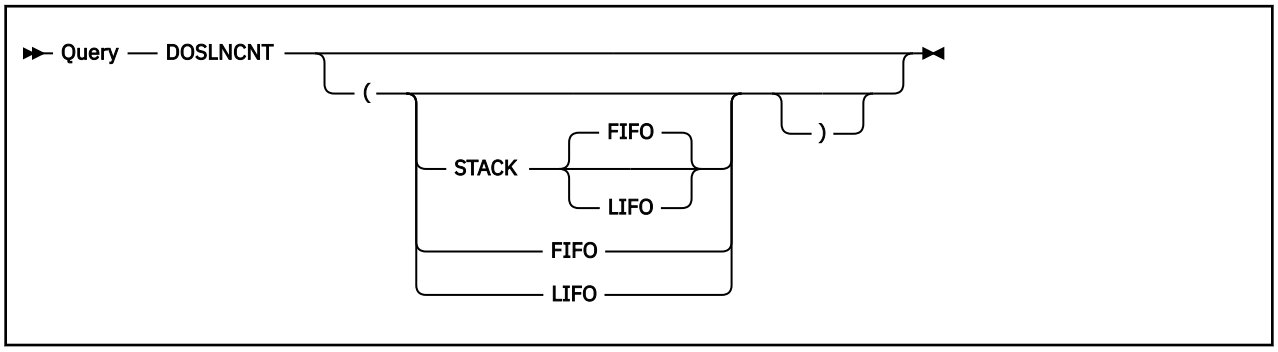
Up to eight names are displayed per line, for as many lines as are necessary. If no DOSLIBs are to be searched, the response is:

```
DOSLIB = NONE
```

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see “QUERY” on page 620.

QUERY DOSLNCNT



Authorization

General User

Purpose

Use the QUERY DOSLNCNT command to display the number of SYSLST lines per page. For more information on how to change the number of SYSLST lines, see “SET DOSLNCNT” on page 925.

Responses

```
DOSLNCNT = nn
```

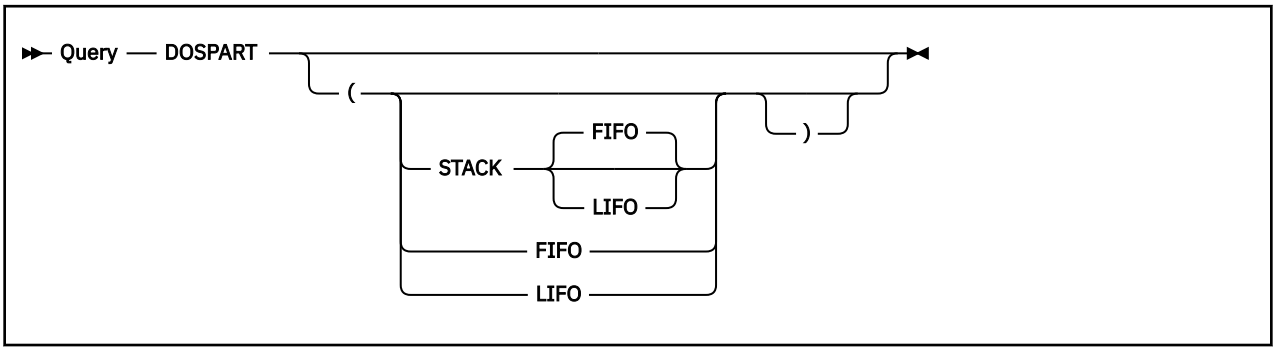
Where:

nn
is an integer from 30 to 99.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see “QUERY” on page 620.

QUERY DOSPART



Authorization

General User

Purpose

Use the QUERY DOSPART command to display the current setting of the virtual partition size. Use the SET DOSPART command to change the partition size. For more information, see "SET DOSPART" on page 926.

Responses

```
nnnnnK
```

or

```
NONE
```

Where:

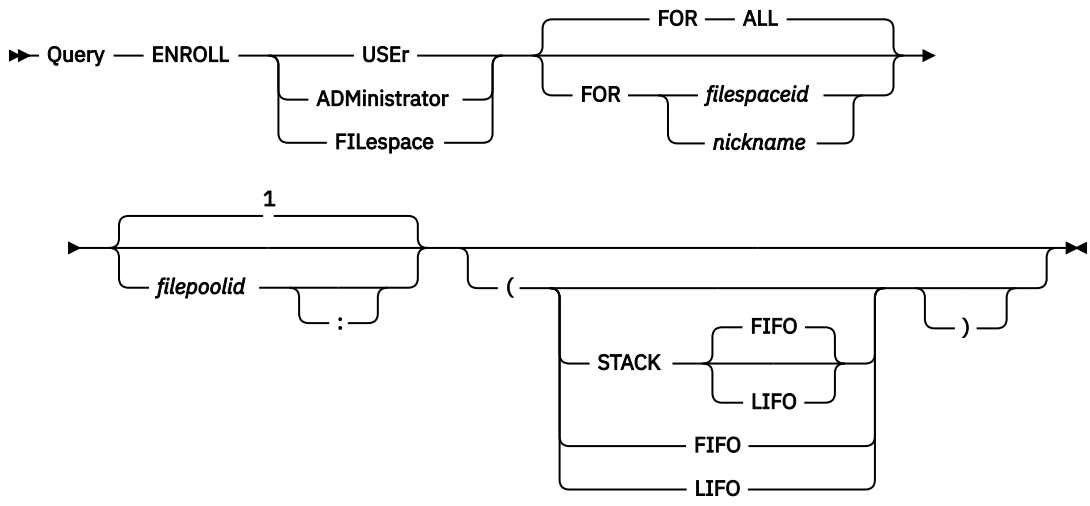
***nnnnn*K**
indicates the size of the virtual partition to be used at program execution time.

NONE
indicates CMS determines the virtual partition size at program execution time.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see "QUERY" on page 620.

QUERY ENROLL



Notes:

[1] If *filepoolid* is not specified, the default file pool ID for your virtual machine is used. If there is no default file pool ID, the command fails.

Authorization

General User

Purpose

Use the QUERY ENROLL command to display information about enrolled users, byte file systems, or administrators in a specified file pool.

Note: If the QUERY ENROLL command is issued from an exec or assembler program for a file pool that is active on the specified work unit, the command will fail.

Operands

USER
specifies user enrollment in the specified file pool is being queried.

ADMINISTRATOR
specifies administrator enrollment in the specified file pool is being queried.

FILESPACE
specifies file space enrollment in the specified file pool is being queried.

FOR ALL
specifies information about all enrolled users, administrators, or file spaces in the specified file pool is to be displayed. This is the default.

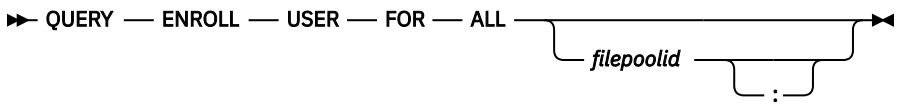
FOR *filespaceid*
specifies a file space ID that identifies a user or byte file system whose enrollment is being queried.

FOR *nickname*
specifies a nickname that represents a file space ID or group of file space IDs whose enrollment is being queried. (Use the NAMES command to define nicknames.)

filepoolid
filepoolid:
specifies the ID of the file pool whose enrollment is being queried. If *filepoolid* is not specified, the default file pool is used. The colon is optional.

Responses

- For




```
Number Of Enrolled Users = nnnn
filespaceid
  .
  .
  .
```

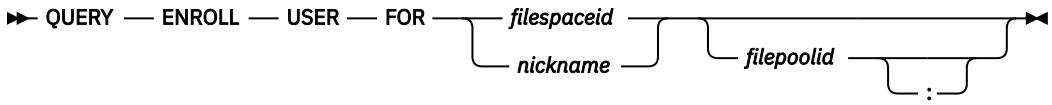
Note: If the file pool administrator has issued a public enrollment for the file pool, <PUBLIC> is displayed as one of the enrolled users.

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.

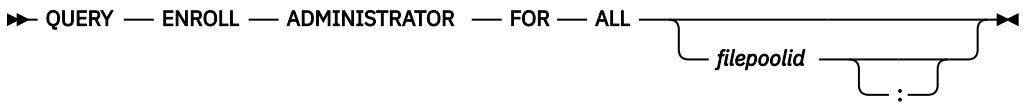
- For




```
Userid        Enrolled
filespaceid   YES/NO
  .             .
  .             .
  .             .
```

Note: The header is generated only if output is displayed at the terminal.

- For



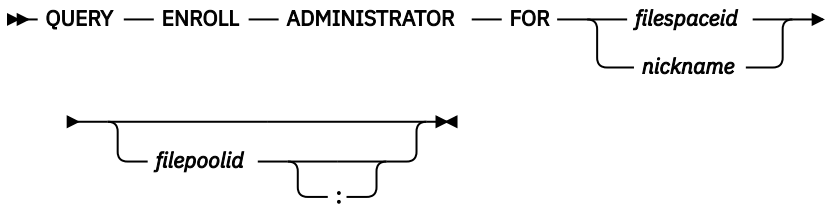

```
Number Of Administrators = nn
filespaceid
  .
  .
  .
```

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.

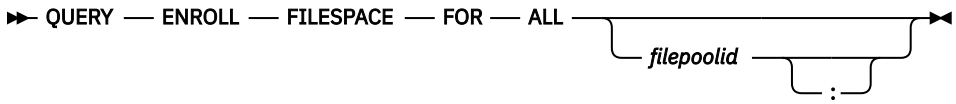
- For




```
Userid        Administrator
filespaceid   YES/NO
  .             .
  .             .
  .             .
```

Note: The header is generated only if output is displayed at the terminal.

- For

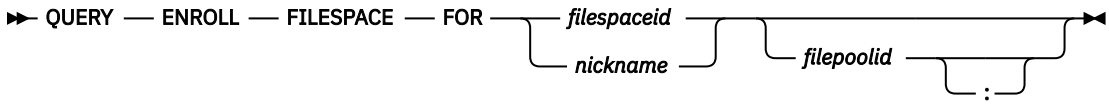



```
Number of file spaces = nnnn
filespaceid     SFS/BFS
  .               .
  .               .
  .               .
```

Note: If there are no file spaces enrolled, you will get this response:

```
No file spaces are enrolled in file pool filepoolid
```

- For




```
Filespace     Enrolled     Type
filespaceid   YES/NO       SFS/BFS
  .              .           .
  .              .           .
```

Note: The header is generated only if output is displayed at the terminal.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see “QUERY” on page 620.

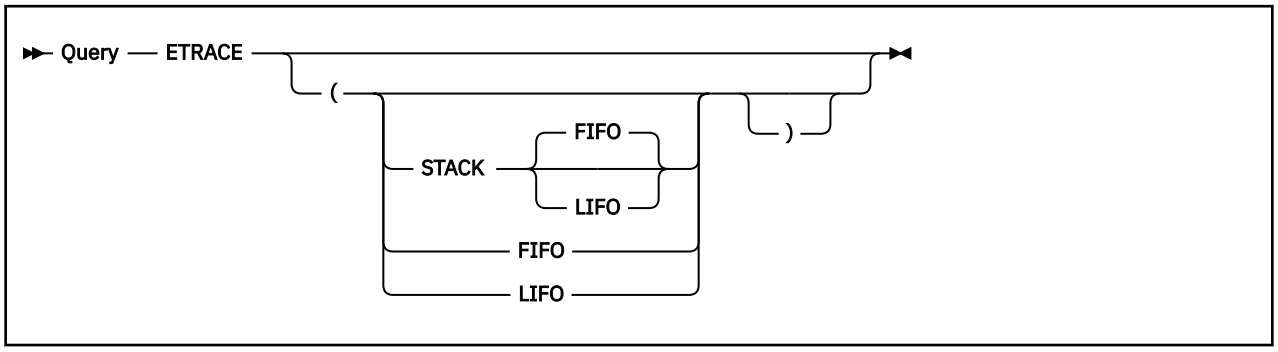
Usage Notes

1. The following are valid ways to query the enrolled users:
   - QUERY ENROLL USER
   - QUERY ENROLL USER FOR ALL
   - QUERY ENROLL USER *filepoolid*
   - QUERY ENROLL USER FOR ALL *filepoolid*
2. The following are valid ways to query the file pool administrators:
   - QUERY ENROLL ADMINISTRATOR

- QUERY ENROLL ADMINISTRATOR FOR ALL
- QUERY ENROLL ADMINISTRATOR *filepoolid*
- QUERY ENROLL ADMINISTRATOR FOR ALL *filepoolid*

3. The following are valid ways to query the enrolled byte file systems:
   - QUERY ENROLL FILESPACE
   - QUERY ENROLL FILESPACE FOR ALL
   - QUERY ENROLL FILESPACE *filepoolid*
   - QUERY ENROLL FILESPACE FOR ALL *filepoolid*

QUERY ETRACE



Authorization

General User

Purpose

Use the QUERY ETRACE command to display a list of the events that are enabled for external tracing.

Responses

```
ETRACE     DMSTRACE
```

or

```
ETRACE     END
```

Where:

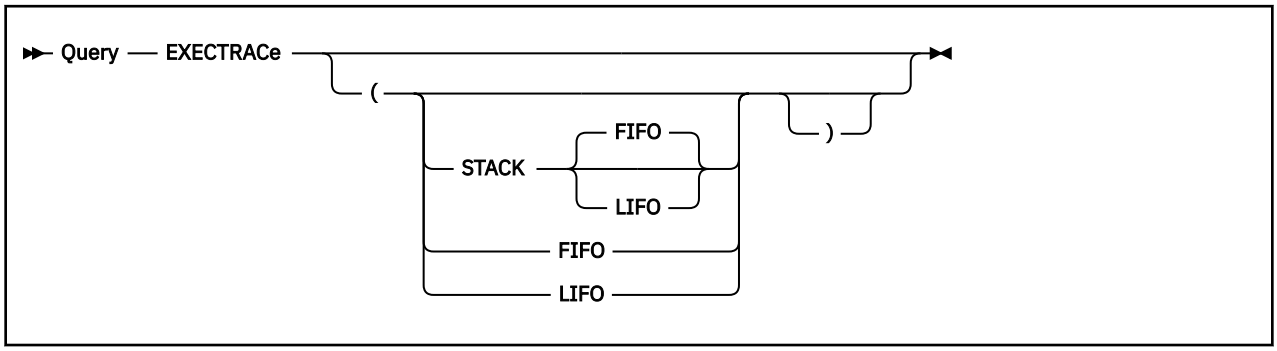
DMSTRACE
indicates external event tracing is enabled

END
indicates external event tracing is disabled.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see “QUERY” on page 620.

QUERY EXECTRACE



Authorization

General User

Purpose

Use the QUERY EXECTRACE command to display the status of exec tracing. Use the SET EXECTRAC command to set the tracing on and off. For more information, see “SET EXECTRACE” on page 928.

Responses

```
EXECTRAC =   ON
```

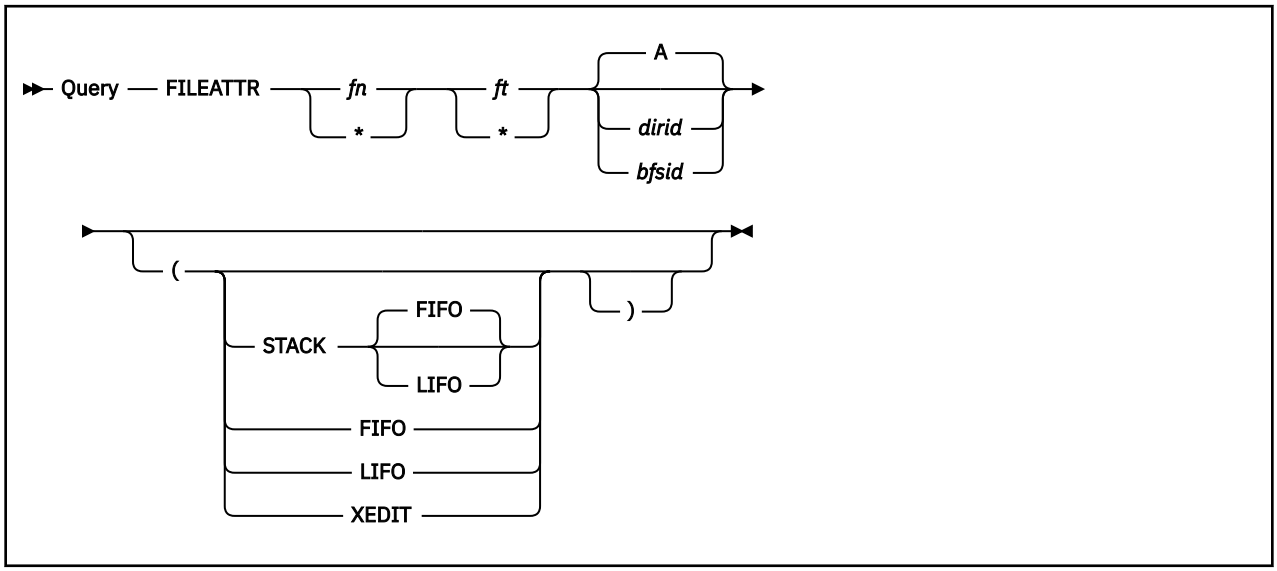
or

```
EXECTRAC =   OFF
```

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see “QUERY” on page 620.

QUERY FILEATTR



Authorization

General User

Purpose

Use the QUERY FILEATTR command to display the recoverability and overwrite attributes of file(s) residing in an SFS directory or byte file system (BFS). For more information on these attributes, see "FILEATTR" on page 263. These attributes may not be changed for BFS files.

Operands

fn
is the file name for which you are requesting attributes.

ft
is the file type queried. If *fn ft* are specified as asterisks, all the attributes of all accessed files in an SFS directory are displayed.

dirid
is the directory identifier being searched. You can also use a file mode for the *dirid* if the directory is already accessed. When no dirid is specified, the file is assumed to be on the directory accessed as A.

bfsid
is the name of the byte file system (BFS) to be searched.

For a BFS file, *fn* and *ft* are system-generated values that together uniquely identify the file within the BFS.

Responses

To receive a response you must have read authority for the specified file(s) or the directory containing the file.

If the file is an alias, the alias is displayed with the attributes of the base file. Assuming all this, a sample response follows for QUERY FILEATTR * * PHILLIPS.PHILO:

```
Directory = FP3:PHILLIPS.PHILO
Filename Filetype Fm Type    Recovery  Overwrite
```

```
PHILLIPS FILELIST A0 BASE    RECOVER   NOTINPLACE
EXAMPLE  SCRIPT   A1 BASE    RECOVER   NOTINPLACE
CODING   STANDRDS A0 ALIAS   NORECOVER NOTINPLACE
CODING1  STANDRDS A0 BASE    NORECOVER INPLACE
PHILLIPS NAMESMN  A0 BASE    RECOVER   NOTINPLACE
PHILLIPS RDRLIST  A0 BASE    RECOVER   NOTINPLACE
PHILLIP1 JMSGSKEL A2 BASE    RECOVER   NOTINPLACE
PHILO2            A  DIR     -         -
PHILO3            A  DIR     -         -
SUBDIR2           A  DIRC    -         -
CODING3  STANDRDS A6 ERASED  -         -
CODING4  STANDRDS A0 REVOKED -         -
MACRO    PREVIOUS A0 ALIAS*  NORECOVER NOTINPLACE
MACRO    HISTORY  A0 BASE*   NORECOVER INPLACE
OTHER    STANDRDS A0 EXTRNL  -         -
Ready;
```

Where:

Filename
is the name of the file or subdirectory.

Filetype
is the file type.

Fm
is the file mode of the directory in which the file or subdirectory resides. If the directory is not accessed, this column contains a dash.

Type
is the type of file or directory as follows:

ALIAS
Alias for a base file

BASE
B file

DIR
Directory having the file control (FILECONTROL) attribute

DIRC
Directory having the directory control (DIRCONTROL) attribute

ERASED
Erased alias (the base file has been erased)

REVOKED
Revoked alias (authority to the base file has been revoked)

ALIAS*
Alias pointing to a base file that is in DFSMS/VM migrated status

BASE*
Base file that is in DFSMS/VM migrated status

EXTRNL
External object

Note: BASE applies only to BFS files (not directories or other BFS objects).

Recovery
is the file's recoverability attribute, as follows:

RECOVER
the file is recoverable

NORECOVER
the file is not recoverable

For subdirectories, this column contains a dash.

Overwrite
is the file's overwrite attribute, as follows:

INPLACE
the file is updated in place

NOTINPLACE
the file is not updated in place

For subdirectories, erased and revoked aliases, and external objects, this column contains a dash.

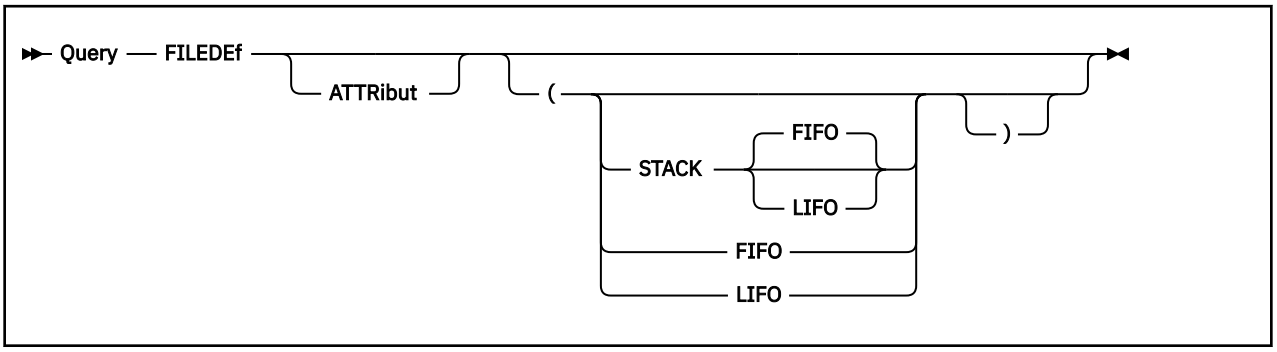
Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see "QUERY" on page 620.

Usage Notes

When using the XEDIT option, the edited file must either be variable length format or fixed format with a record length of 49.

QUERY FILEDEF



Authorization

General User

Purpose

Use the QUERY FILEDEF command to display all the file definitions in effect.

Use the FILEDEF command to set file definitions. For more information, see "FILEDEF" on page 266.

Operands

ATTRibut
requests the display of the RECFM, LRECL, and BLOCK size attributes associated with this FILEDEF.

Responses

```
ddname device  (additional information by device type)
  .      .            .
  .      .            .
  .      .            .
```

Just the *ddname* and *device* are displayed for device types TERMINAL, PRINTER, PUNCH, READER, and DUMMY. Additional information is displayed for the following device types:

```
ddname DISK  filename|vdev
 filetype filemode [datasetname]

ddname TAPn  labeltype [n [VOLID volid]|filename]

ddname GRAF  vaddr

ddname VSAM  filename filetype emulatorname
```

Note: For information on specifying alternate VSAM emulators, see *SQL/DS Application Interface for VSAM*.

If the ATTRIBUT operand is specified, the following line is also displayed:

```
RECFM recfm LRECL nnnnn BLOCK nnnnn
```

If no file definitions are in effect, the following message is displayed at the terminal:

```
No user defined FILEDEFs in effect
```

Examples

1. John wants to display all his file definitions that are in effect, so he enters:

```
query filedef
```

and the following is displayed:

```
MINE    DISK      TEST     SAMPLE     A1
FILEA   TAP2   SL  00002  VOLID  DEPT10
IN      TAP6   LABOFF
OUT     TAP8   LABOFF
```

These file definitions were set with the following commands:

```
filedef mine disk test sample a1
filedef filea tap2 sl 2 volid dept10
filedef in tap6 (18TRACK
filedef out tap8 (18TRACK
```

2. If Mary enters the following FILEDEF:

```
filedef syslib disk mysys loadlib a (recfm v lrecl 13030
```

and then issues a query with attributes:

```
query filedef attr
```

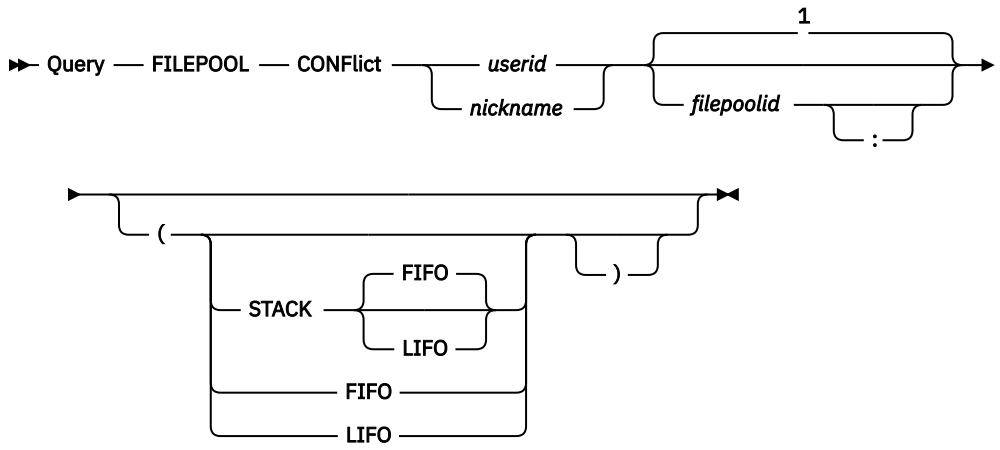
the following is displayed:

```
SYSLIB   DISK     MYSYS     LOADLIB  A1
      RECFM V    LRECL 13030 BLOCK 00000
```

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see “QUERY” on page 620.

QUERY FILEPOOL CONFLICT



Notes:

[1] The default is the default file pool ID for your virtual machine. If there is no default file pool ID, the command fails.

Authorization

General User

Purpose

Use the QUERY FILEPOOL CONFLICT command to display information about lock conflicts in the specified file pool for the specified user ID or set of users. The information returned can assist you with the task of diagnosing why file pool users are experiencing delays in file pool server responses. For more information specify a nickname for a set of users, see "NAMES" on page 535.

Operands

userid
specifies the user whose lock conflicts to the specified file pool is to be displayed.

nickname
identifies a nickname that represents a user ID or a list of user IDs. (Use the NAMES command to define nicknames.)

filepoolid
filepoolid:
identifies the file pool. If not specified, the default file pool is used. The colon is optional.

Responses

```
Requester   Holder   Wait    Lock   Lock Type
requester   holder   wait    lock   lock type
   .          .       .       .       .
   .          .       .       .       .
   .          .       .       .       .
```

Note:

1. The header is generated only if output is displayed at the terminal.

2. You cannot use QUERY FILEPOOL CONFlict to find your own conflicts. If you receive a response, you have no conflicts. When you are in a lock wait state, for example, a file pool conflict, your virtual machine waits for the file pool server to reply and you cannot enter any commands until the conflict is resolved.
3. If you have file pool administration authority, an additional column of information is returned. For more about the administrator's version of this command, see *z/VM: CMS File Pool Planning, Administration, and Operation*.

Where:

requester
is the user who is waiting for a lock that is being held by the holder.

holder
is the user who is preventing the requester 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.

AVSwait
is displayed when the CRR recovery server agent is waiting for a response from AVS. The AVS virtual machine is the agent's communication partner.

Catalog_Buffer
means the holder of the lock is waiting for a buffer to hold catalog minidisk data.

Checkpoint
means the holder of the lock is waiting for an internal checkpoint to complete.

Communication
means the file pool server is waiting for another request from the holder of the lock.

Control_Buffer
means the holder of the lock is waiting for a buffer to hold control minidisk data.

DFSMS/VM_Wait
is displayed when the agent is waiting for a DFSMS/VM response.

ESM_Wait
is displayed when the agent is waiting for an External Security Manager (ESM) response.

I/O
means the holder of the lock is waiting for file pool I/O

Lock
means 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.

Multi_Event
is displayed when the CRR recovery server communicates with multiple participating resource managers (such as the SFS file pool server) and waits for a response from a participating resource manager.

None
there is no wait state.

Storage_Wait
is displayed when the agent is waiting for storage to be available on the server.

This wait state applies to all file pool servers.

Subtask
is displayed when the agent sends a request to the file pool server, which must be suspended pending the completion of a file pool function (for example, the initial exchange of log names with the user's CRR recovery server). The file pool function is handled as a subtask, which is not directly related to the user's request that resulted in the subtask being initiated. While the subtask is executing, the original user's request is suspended in this wait state.

Suspended
is displayed when the CRR recovery server agent is temporarily put into a waiting state by the CRR SUSPEND operator command. This wait state prevents the periodic retry of resynchronization.

Timer
is displayed when the CRR recovery server agent is waiting for a time period to elapse (as specified by the RESYNCINTERVAL start-up parameter or defaults to 10 minutes) before the periodic retry of resynchronization in an attempt to complete the update of a protected resource.

Token_Callback
is displayed when the file pool server is waiting for the byte file system (BFS) client machine to respond to a request to not validate the cached file pool information.

—
is displayed when the file pool server is at a higher release level than the CMS in your machine and the wait is a type that is unknown.

lock
is one of the following types of file pool resources for which the Requestor has requested a lock.

BFS_BR_Lock
BFS_Cre/Del
BFS_Glob_Stor
BFS_Name_Unal
BFS_Obj_Write
Catalog_Block
Catalog_Index
Catalog_Row
DIRC_Resource (denotes a DIRCONTROL resource)
Directory
File
File_Recall (recall of a DFSMS/VM migrated file)
File Space
Recall_Exit (recall of a DFSMS/VM migrated file)
Recovery
Storage_Group
—

lock type
is the type of lock the requester needs. Some of these are internal lock types 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.

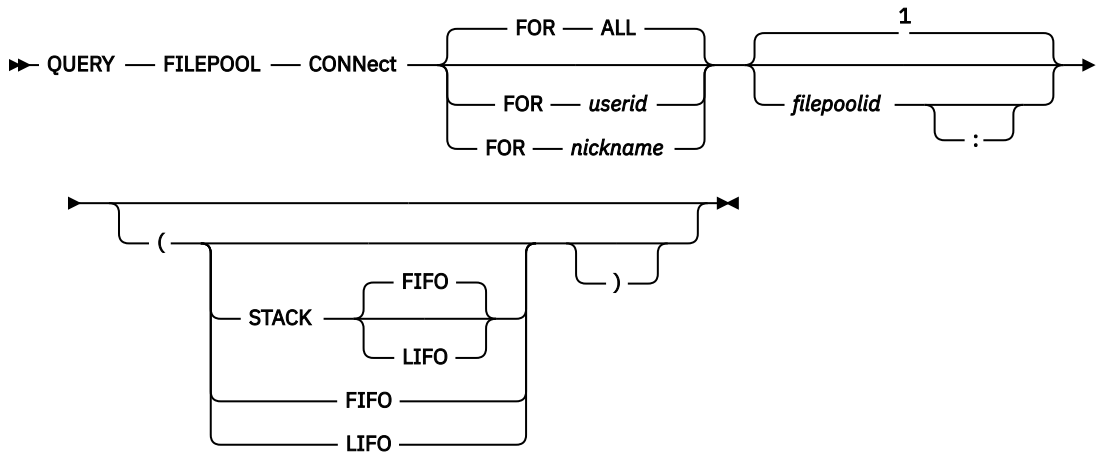
Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see “QUERY” on page 620.

Usage Notes

You cannot use this command to find lock conflicts caused by byte range locking of byte files.

QUERY FILEPOOL CONNECT



Notes:

[1] The default is the default file pool ID for your virtual machine. If there is no default file pool ID, the command fails.

Authorization

General User

Purpose

Use the QUERY FILEPOOL CONNECT command to display information about who is connected to a file pool.

Operands

FOR ALL
means information for all users is to be displayed. This is the default.

FOR *userid*
specifies the user whose connection to the specified file pool is to be displayed.

FOR *nickname*
identifies a nickname that represents a user ID or a list of user IDs. (Use the NAMES command to define nicknames.)

filepoolid
filepoolid:
identifies the file pool. If not specified, the default file pool is used. The colon is optional.

Responses

```
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 or is @RESYNC. @RESYNC is used for the Coordinated Resource Recovery (CRR) facility. For more information about SFS participation in CRR, see *z/VM: CMS File Pool Planning, Administration, and Operation*. @RESYNC is a special user ID for the CRR recovery server, which performs CRR resynchronization functions.

Note: A single user ID may have multiple connections and would be listed for each connection.

connected
is one of the following:

Active =
connected to the file pool for SFS activity and has an active logical unit of work.

Active_BFS
connected to the file pool for byte file system (BFS) activity and has an active file pool server request.

Active_Syncpt_Logr =
CRR logger connections (not distinguished from normal user connections).

No =
not connected (not applicable if ALL is specified)

Prepared&Connected =
connected but waiting for the second phase of a two-phase commit.

Prepared&No_Connect =
not connected; communication was severed after the first phase of a two-phase commit.

Yes =
connected to the file pool for file pool activity and does not have an active logical unit of work.

Yes_BFS
connected to the file pool for byte file system (BFS) activity and does not have an active file pool server request.

Yes_Incoming_AVS =
AVS control connections supporting CRR resynchronization.

Yes_Incoming_Resync =
incoming resynchronization connections to:

- the CRR recovery server from another CRR recovery server
- the CRR recovery server from a participating resource manager
- the participating SFS resource manager (server) from the CRR recovery server

Yes_Outgoing_AVS =
AVS control connections supporting CRR resynchronization.

Yes_Outgoing_Resync =
various outgoing connections are initiated by CRR resynchronization.

Yes_Syncpt_Logr =
CRR logger connections (not distinguished from normal user connections).

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see "QUERY" on page 620.

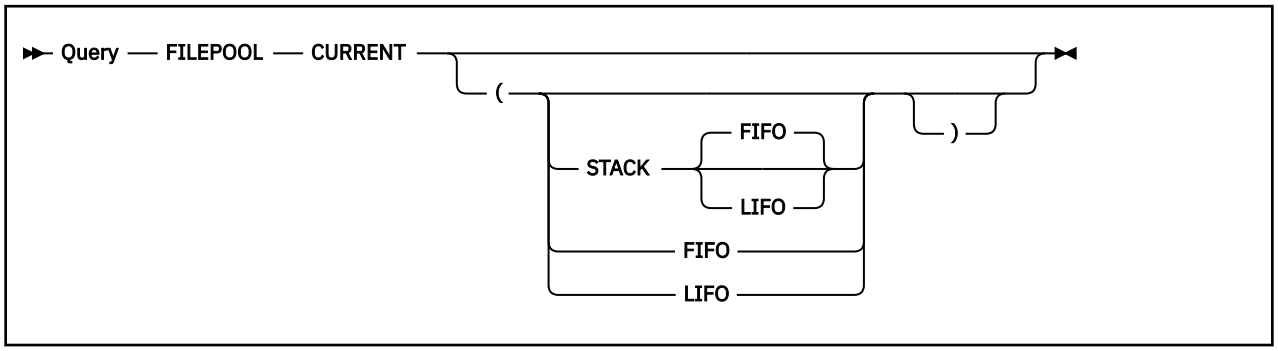
Usage Notes

These are the valid ways to query the connected users:

- QUERY FILEPOOL CONNECT
- QUERY FILEPOOL CONNECT FOR ALL

- QUERY FILEPOOL CONNECT *filepoolid*
- QUERY FILEPOOL CONNECT FOR ALL *filepoolid*

QUERY FILEPOOL CURRENT



Authorization

General User

Purpose

Use the QUERY FILEPOOL CURRENT command to display the current default file pool set by the SET FILEPOOL command.

Responses

```
FILEPOOL filepoolid:
```

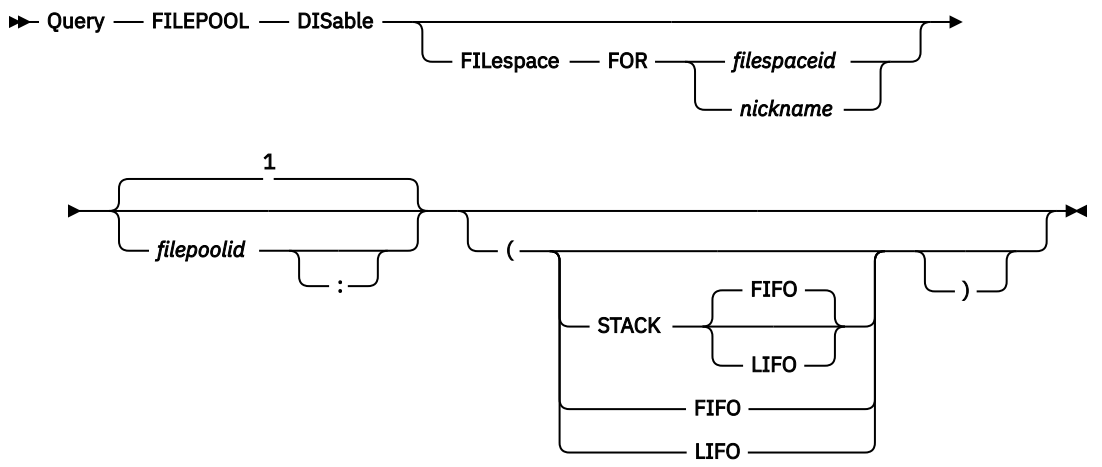
If there is no current file pool, the response is:

```
FILEPOOL NONE
```

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see “QUERY” on page 620.

QUERY FILEPOOL DISABLE



Notes:

[1] The default is the default file pool ID for your virtual machine. If there is no default file pool ID, the command fails.

Authorization

General User

Purpose

Use the QUERY FILEPOOL DISABLE command to determine disable lock information. There are two versions of this command:

1. Use QUERY FILEPOOL DISABLE with no keyword specified to determine if your file space or its owning storage group is disabled. For more information, see Usage Note "2" on page 697.
2. Use QUERY FILEPOOL DISABLE FILESPACE to determine disable lock information for a file space or its owning storage group.

   Any file pool users can enter this command on a file space for their file space ID, another user's file space, or for a byte file system (BFS) enrolled in the file pool.

Note: A different version of this command is available for users with file pool administration authority. For more information, see the QUERY FILEPOOL DISABLE command in *z/VM: CMS File Pool Planning, Administration, and Operation*.

Operands

FILespace FOR *filespaceid*
identifies the name of a file space for which you want information.

FILespace FOR *nickname*
identifies a nickname that represents the name of a file space. This can be a nickname for a single user ID or a nickname that represents a byte file system. The nickname must be a single name; it must not represent a list of names. For more information on defining nicknames, see "NAMES" on page 535.

filepoolid
filepoolid:
identifies the file pool being queried. If not specified, the default file pool is used. The colon is optional.

Responses

A single response or multiple responses might be displayed:

```
Object Type  Object ID    Disabled Creator   Mode
FILESPACE    filespaceid  YES      userid2   modetype
GROUP        –            YES      userid    modetype
FILESPACE    filespaceid  YES      function  modetype
GROUP        –            YES      function  modetype
```

or, if no disables exist, the response is:

```
Object Type  Object ID    Disabled Creator   Mode
FILESPACE    filespaceid  NO       –         –
```

Note: The header is generated only if output is displayed at the terminal, that is, it is not stacked.

Where:

Object Type
this column shows the type of object. It can be either FILESPACE or GROUP.

Object ID
this column shows the *filespaceid* for object type of FILESPACE.

'–' (dash) is displayed when object type is GROUP.

Disabled
this column shows whether the object (FILESPACE or GROUP) is disabled (YES) or not disabled (NO).

Creator
this column indicates the user ID or function who has the file space or its owning storage group disabled.

– (dash) is displayed when the file space or its owning storage group was not disabled.

Note: If the DISABLE operator command was used to disable the object and an *owner* was specified on that command, the creator's user ID is the *owner,* not the user ID that issued the disable.

Mode
this column indicates the mode. The *modetype* can be one of these:

SHARE
is displayed when a share disable lock exists on the file space or its owning storage group. A share disable lock allows users to read, but not modify, items in the locked object.

EXCLUSIVE
is displayed when a exclusive disable lock exists on the file space or its owning storage group. An exclusive disable lock prevents anyone else but the Creator from reading or modifying the items in the locked object.

IEXCLUSIVE
is displayed when an intent exclusive disable lock exists on the file space or its owning storage group. An intent exclusive disable lock could prevent anyone else but the Creator from reading or modifying the items in the locked object. It is an internal lock.

– (dash)
is displayed when the file space or its owning storage group was not disabled.

*** (asterisk)**
is displayed when the file pool server is at a higher release level than the CMS in your machine, and the mode is a type unknown to CMS.

Examples

The user machine displays responses describing the disable status of the object being queried.

1. For a QUERY FILEPOOL DISABLE, issued from user ID JONES, the output might look like:

```
Object Type  Object ID  Disabled Creator     Mode
GROUP        -          YES      CHERIE      SHARE
FILESPACE    JONES      YES      ANDY        SHARE
```

   this tells you the file space JONES was disabled in share mode by user ANDY. Also, it tells you user CHERIE also disabled the owning storage group in share mode.

2. For a QUERY FILEPOOL DISABLE FILESPACE FOR JONES, the output might look like:

```
Object Type  Object ID  Disabled Creator     Mode
GROUP        -          YES      BROWN       SHARE
FILESPACE    JONES      YES      BROWN       SHARE
FILESPACE    JONES      YES      DOE         SHARE
FILESPACE    JONES      YES      SMITH       SHARE
```

   this tells you file space JONES was disabled by users' BROWN, DOE, and SMITH in share mode and the storage group that owns file space JONES was also disabled in share mode by user BROWN.

   Or,

```
Object Type  Object ID  Disabled Creator     Mode
FILESPACE    JONES      YES      MARY        EXCLUSIVE
```

   this tells you file space JONES was disabled by user MARY in exclusive mode.

   Or,

```
Object Type  Object ID  Disabled Creator     Mode
GROUP        -          YES      MARY        EXCLUSIVE
```

   this tells you file space JONES was not disabled but its owning storage group was disabled in exclusive mode by user MARY.

   Or,

```
Object Type  Object ID  Disabled Creator     Mode
FILESPACE    JONES      NO       -           -
```

   this tells you file space JONES was not disabled and its owning storage group was not disabled.

   Or,

```
Object Type  Object ID  Disabled Creator     Mode
FILESPACE    JONES      YES      RENAME      EXCLUSIVE
GROUP        -          YES      RENAME      IEXCLUSIVE
```

   this tells you file space JONES was disabled in exclusive mode and its owning storage group was disabled in intent exclusive mode. These are internal disables created by the RENAME function. For more information on the RENAME function, see Usage Note “4” on page 697.

Usage Notes

1. This command might be used to determine disable information for a specified file space and/or its owning storage group when the results are lost or unknown of one of the following:
   - DISABLE operator command
   - FILEPOOL DISABLE command
   - Disable File Space (DMSDISFS) CSL routine
   - Disable Storage Group (DMSDISSG) CSL routine
   - FILEPOOL BACKUP command

- FILEPOOL RESTORE command
- FILEPOOL RENAME command

For more information on disable locks on a file space or its owning storage group, see the respective command as shown in Usage Note "1" on page 696, in *z/VM: CMS File Pool Planning, Administration, and Operation*, or see the respective CSL routine as shown in Usage Note "1" on page 696, in *z/VM: CMS Callable Services Reference*.

2. These are the valid ways to query your own disables:
   - QUERY FILEPOOL DISABLE
   - QUERY FILEPOOL DISABLE *filepoolid*
   - QUERY FILEPOOL DISABLE FILESPACE FOR *myuserid*
   - QUERY FILEPOOL DISABLE FILESPACE FOR *myuserid filepoolid*
3. If the QUERY FILEPOOL DISABLE command does not resolve what type of lock exists, there may be an explicit lock on the file or directory.

   For more information on a possible explicit lock on the file or directory, see "QUERY LOCK" on page 726.
4. If the file space and its owning storage group was disabled due to a function, the function that created the disables will be returned.

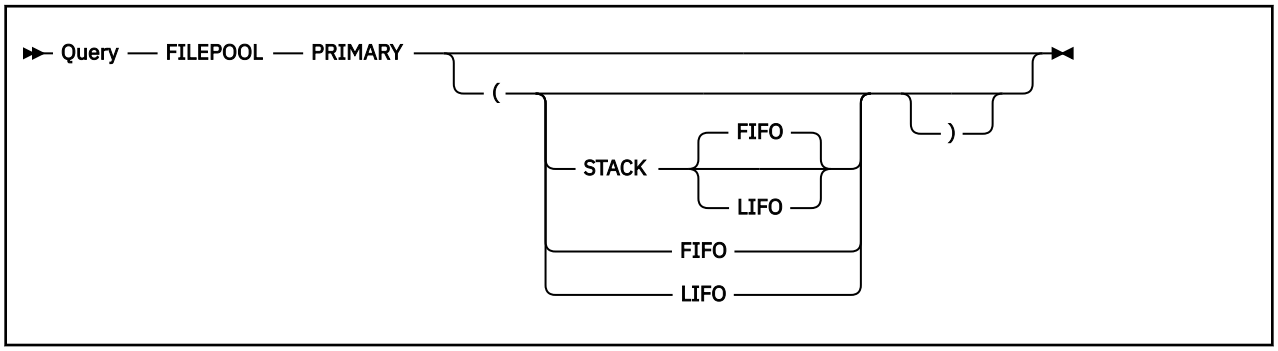
   For example, an attempt of the FILEPOOL RENAME command can cause the file space and its owning storage group to be disabled; in these cases 'RENAME' will be returned as the creator of the disable.

   Note: RENAME is currently the only valid function name. For more information on RENAME, see the Usage Notes of the FILEPOOL RENAME command in *z/VM: CMS File Pool Planning, Administration, and Operation*.
5. If the QUERY FILEPOOL DISABLE command is issued from an exec or an assembler program, and the file pool specified is active (that is, has some work on it which has not been committed or rolled back) on the current default work unit, the command will fail.
6. To get the equivalent information from an application program, use the Query Filepool Disable CSL routine DMSQFPDS. For more information, see *z/VM: CMS Callable Services Reference*.
7. If you specify a nickname and the NODE tag, in the NAMES file, it indicates the user is on another processor. The LOCALID tag must also be specified. For more information, see "NAMES" on page 535.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see "QUERY" on page 620.

QUERY FILEPOOL PRIMARY



Authorization

General User

Purpose

Use the QUERY FILEPOOL PRIMARY command to display the primary file pool specified during IPL.

Responses

```
FILEPOOL filepoolid:
```

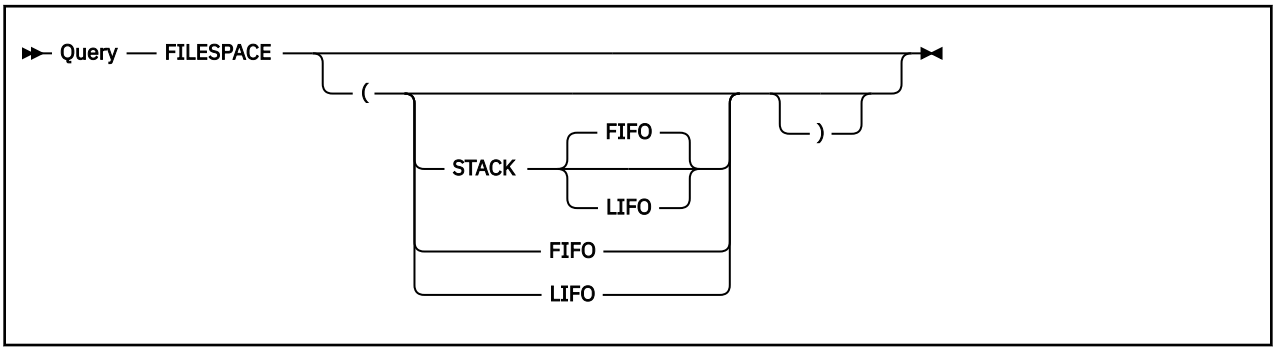
If there is no current file pool, the response is:

```
FILEPOOL NONE
```

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see “QUERY” on page 620.

QUERY FILESPACE



Authorization

General User

Purpose

Use the QUERY FILESPACE command to display information about the default file space. Use the SET FILESPACE command to set the default file space. For more information, see “SET FILESPACE” on page 931.

Responses

```
FILESPACE  userid
```

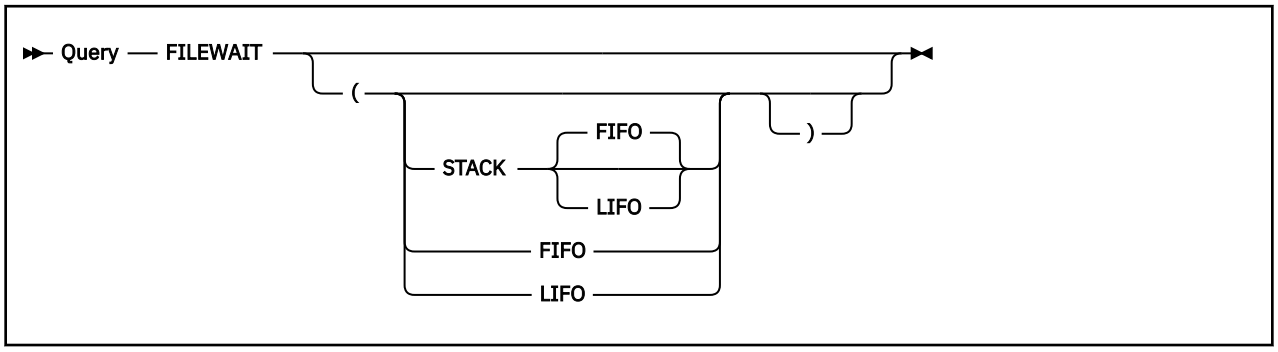
Where:

userid
identifies the default file space ID.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see “QUERY” on page 620.

QUERY FILEWAIT



Authorization

General User

Purpose

Use the QUERY FILEWAIT command to learn if the commands acting on files in a file pool have been set to wait for locked files to become available. For more information, see “SET FILEWAIT” on page 933.

Responses

```
FILEWAIT  ON
```

or

```
FILEWAIT  OFF
```

Where:

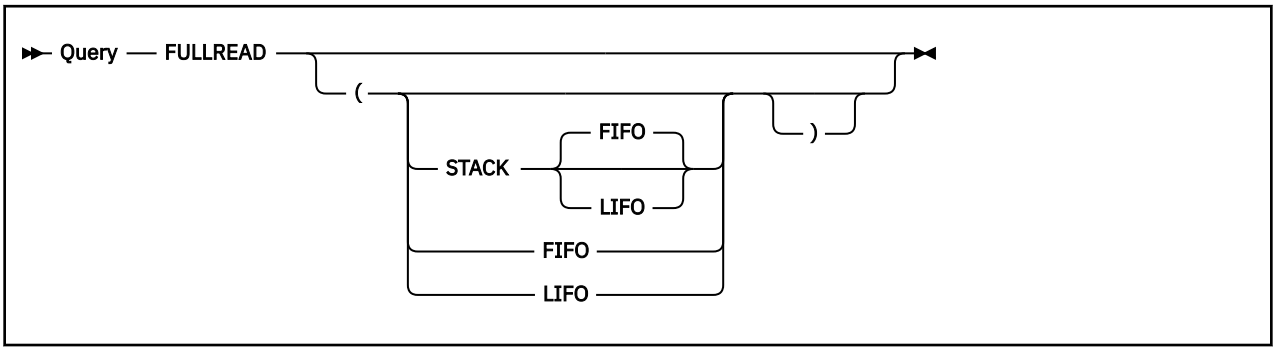
ON
specifies you do not want commands affecting a file to fail because the file is unavailable. The request waits until the file becomes available. The wait may be for a prolonged time.

OFF
specifies commands affecting a file will fail immediately if the requested file is not available.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see “QUERY” on page 620.

QUERY FULLREAD



Authorization

General User

Purpose

Use the QUERY FULLREAD command to display whether the 3270 null characters are recognized in the middle of the physical screen. Use the SET FULLREAD command to change the FULLREAD setting. For more information, see "SET FULLREAD" on page 935.

Responses

```
FULLREAD     ON
```

or

```
FULLREAD     OFF
```

Where:

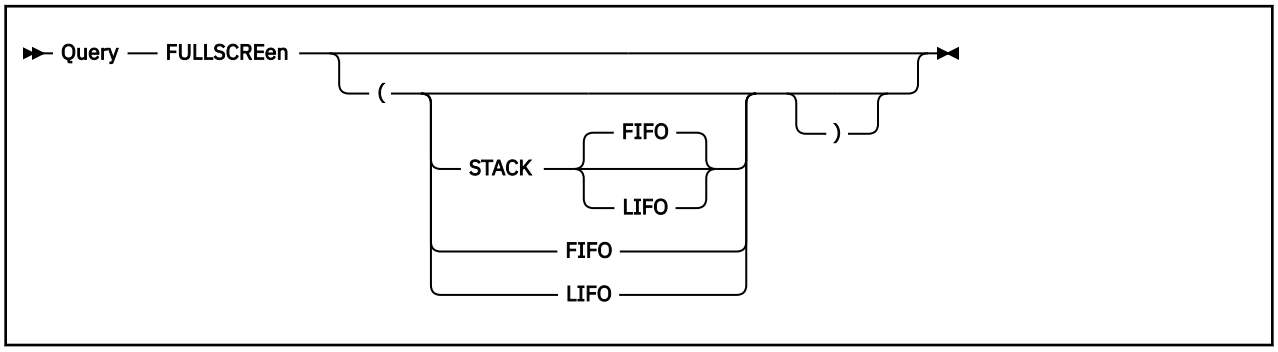
ON
indicates null characters are recognized in the middle of lines, making it easier for you to enter tabular or pictorial data.

OFF
inhibits transmission of nulls from the terminal.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see "QUERY" on page 620.

QUERY FULLSCREEN



Authorization

General User

Purpose

Use the QUERY FULLSCREEN command to display the status of full-screen CMS. For more information on how to change the full-screen setting, see “SET FULLSCREEN” on page 937.

Responses

```
FULLSCREEN ON
```

or

```
FULLSCREEN OFF
```

or

```
FULLSCREEN SUSPEND
```

Where:

ON
indicates full-screen CMS is active.

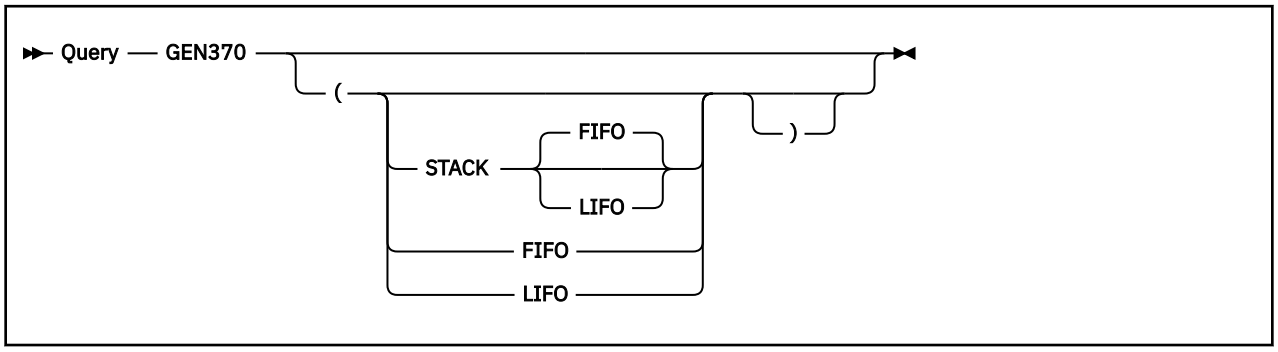
OFF
indicates full-screen CMS is inactive.

SUSPEND
indicates full-screen CMS is suspended.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see “QUERY” on page 620.

QUERY GEN370



Authorization

General User

Purpose

Use the QUERY GEN370 command to determine the current value of the CMS SET GEN370 command.

Responses

```
GEN370  =  ON
```

or

```
GEN370  =  OFF
```

Where:

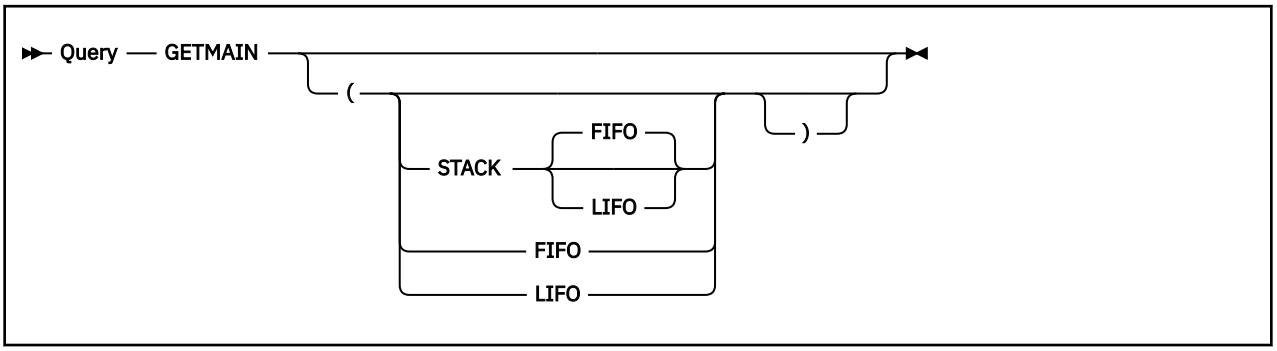
ON
means modules generated with the GENMOD 370 option will not be executed in an XA or XC virtual machine. This is the initial setting.

OFF
means modules generated with the GENMOD 370 option will be executed in an XA or XC virtual machine.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see “QUERY” on page 620.

QUERY GETMAIN



Authorization

General User

Purpose

Use the QUERY GETMAIN command to display the current setting of the SET GETMAIN command. Use the SET GETMAIN command to control the amount of storage obtained on variable GETMAIN storage requests. For more information, see "SET GETMAIN" on page 946.

Responses

```
GETMAIN = PERCENT nnn
```

or

```
GETMAIN = OFF
```

Where:

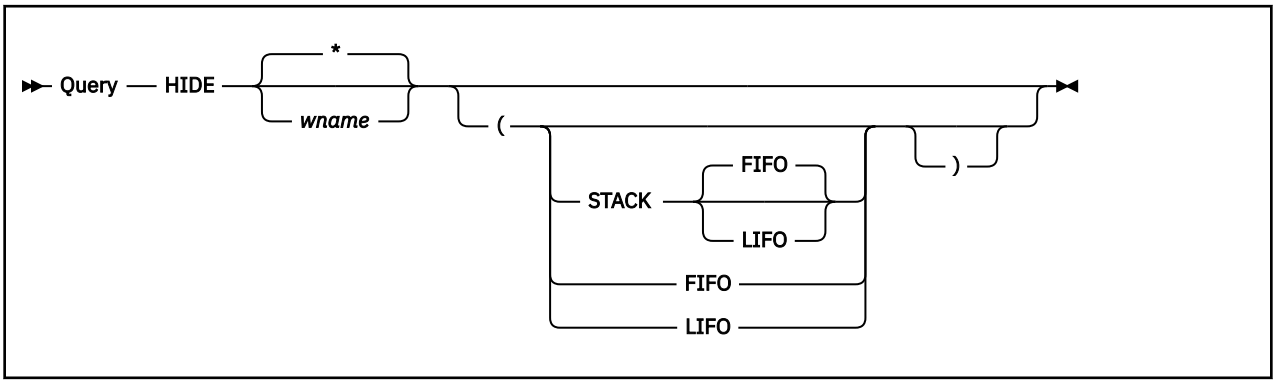
PERCENT *nnn*
indicates the maximum amount of storage that can be obtained for a variable GETMAIN request is *nnn* percent of the largest contiguous piece of storage available. The variable *nnn* is a decimal number from 1 to 100.

OFF
indicates the maximum amount of storage that can be obtained for a variable GETMAIN request is based on the size of the virtual machine, as follows:

VMSIZE	Percentage (approx.)
less than or equal to 4MB	67%
5MB/6MB	80%
greater than 6MB	95%

This is the default value.

QUERY HIDE



Authorization

General User

Purpose

Use the QUERY HIDE command to display information about one or all hidden windows.

Operands

wname
is the name of the window about information is to be displayed.

***
indicates information about all hidden windows is to be displayed. This is the default.

Responses

One line is displayed per hidden window. If the specified window is not displayed, you will simply get a message.

- `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.

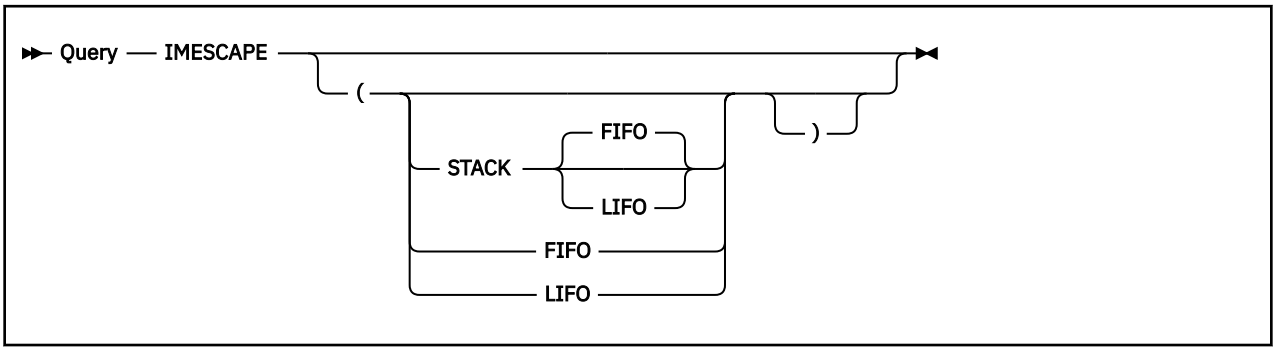
  line
  is the line number in the virtual screen where the window is connected.

  column
  is the column number in the virtual screen where the window is connected.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see “QUERY” on page 620.

QUERY IMESCAPE



Authorization

General User

Purpose

Use the QUERY IMESCAPE command to display the immediate command escape character. Use the SET IMESCAPE command to change the character. For more information, see "SET IMESCAPE" on page 948.

Responses

```
IMESCAPE = char
```

or

```
IMESCAPE = OFF
```

Where:

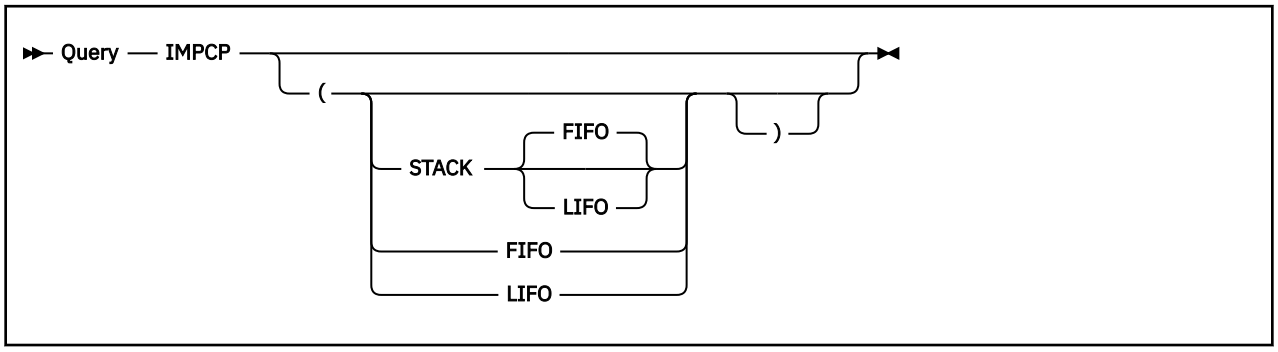
char
is the escape character in effect. The default character is a semi-colon (;).

OFF
no immediate command escape character is in effect.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see "QUERY" on page 620.

QUERY IMPCP



Authorization

General User

Purpose

Use the QUERY IMPCP command to display the status of the implied CP command indicator. Use the SET IMPCP command to change the IMPCP setting. For more information, see "SET IMPCP" on page 949.

Responses

```
IMPCP    = ON
```

or

```
IMPCP    = OFF
```

Where:

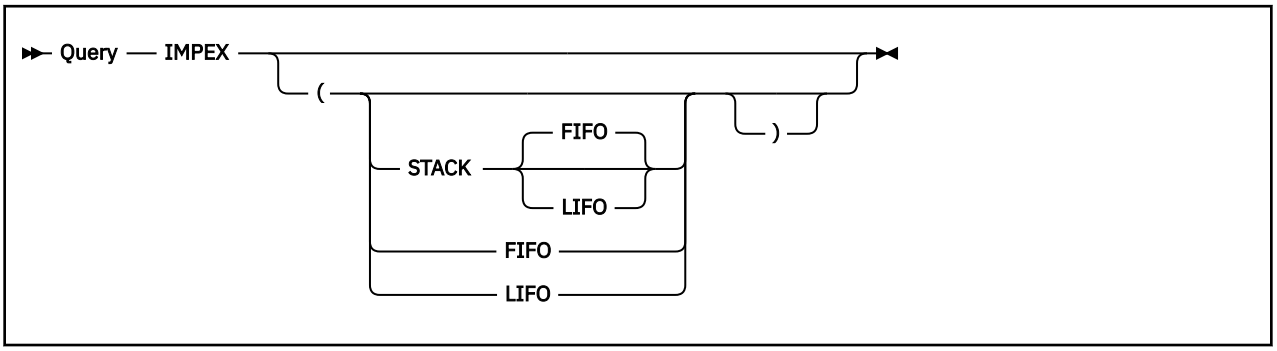
ON
indicates commands CMS does not recognize are passed to CP; that is, unknown commands are considered to be CP commands.

OFF
indicates an error message will be generated at the terminal if a command is not recognized by CMS. You must use the CMS CP command to execute CP commands from the CMS environment.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see "QUERY" on page 620.

QUERY IMPEX



Authorization

General User

Purpose

Use the QUERY IMPEX command to display the status of the implied exec indicator. Use the SET IMPEX command to change the setting of the indicator. For more information, see “SET IMPEX” on page 950.

Responses

```
IMPEX    = ON
```

or

```
IMPEX    = OFF
```

Where:

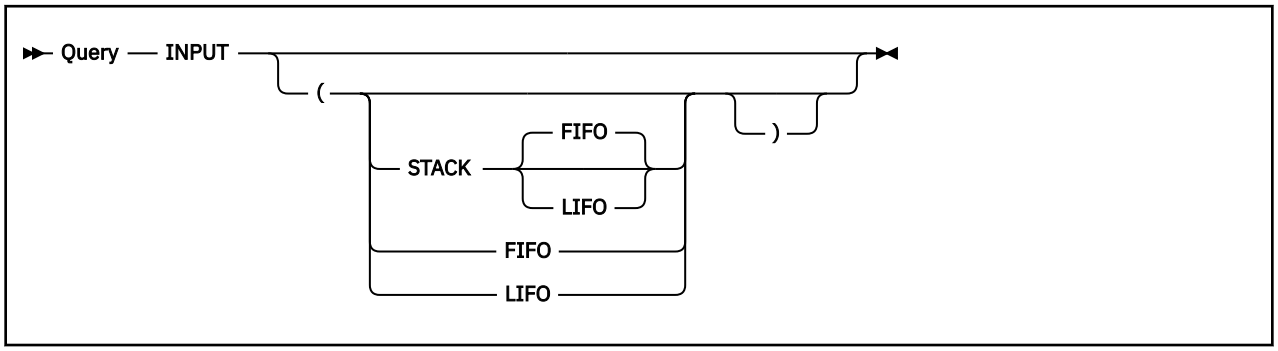
ON
indicates exec files can be executed by entering the file name of the file.

OFF
indicates the EXEC command must be explicitly entered to execute exec files.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see “QUERY” on page 620.

QUERY INPUT



Authorization

General User

Purpose

Use the QUERY INPUT command to display the contents of any input translation table in effect. Use the SET INPUT command to set up input translation tables. For more information, see "SET INPUT" on page 951.

Responses

```
INPUT = a1  xx1
        .   .
        .   .
        .   .
        an  xxn
```

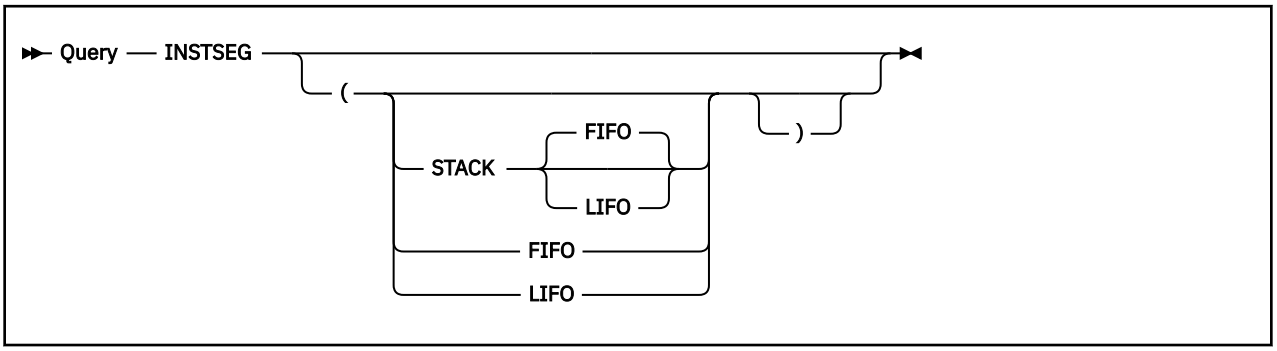
If you do not have an input translate table in effect, the response is:

```
No user defined input translate table in use
```

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see "QUERY" on page 620.

QUERY INSTSEG



Authorization

General User

Purpose

Use the QUERY INSTSEG command to display the status of the CMS installation saved segment. Use the SET INSTSEG command to change the INSTSEG setting. For more information, see "SET INSTSEG" on page 952.

Responses

```
INSTSEG  = ON mode|LAST
```

or

```
INSTSEG  = OFF
```

Where:

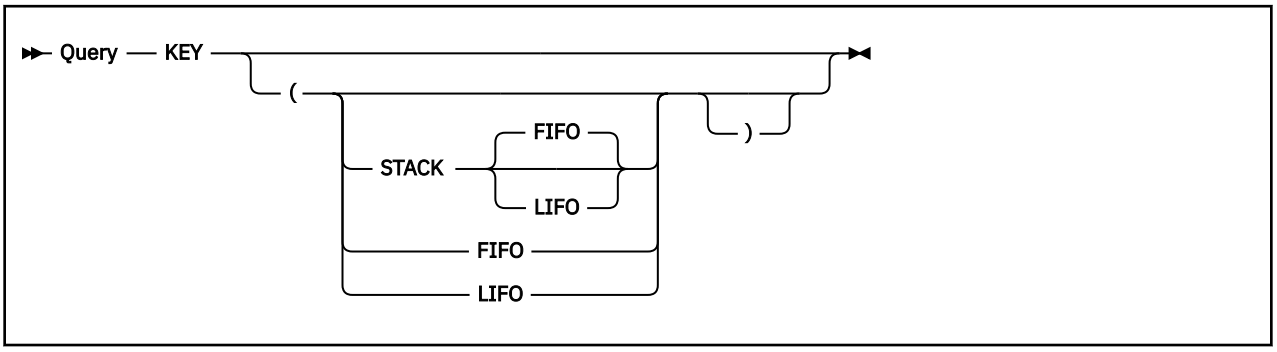
ON mode|LAST
indicates the installation saved segment is being used. The *mode* specifies the location in the command search order where the saved segment is being accessed. LAST indicates the CMS installation saved segment is searched after the last accessed disk or directory in the search order.

OFF
indicates the CMS installation saved segment is not being used.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see "QUERY" on page 620.

QUERY KEY



Authorization

General User

Purpose

Use the QUERY KEY command to display the last key pressed that caused an attention interrupt.

Responses

```
KEY    =    keypressed
```

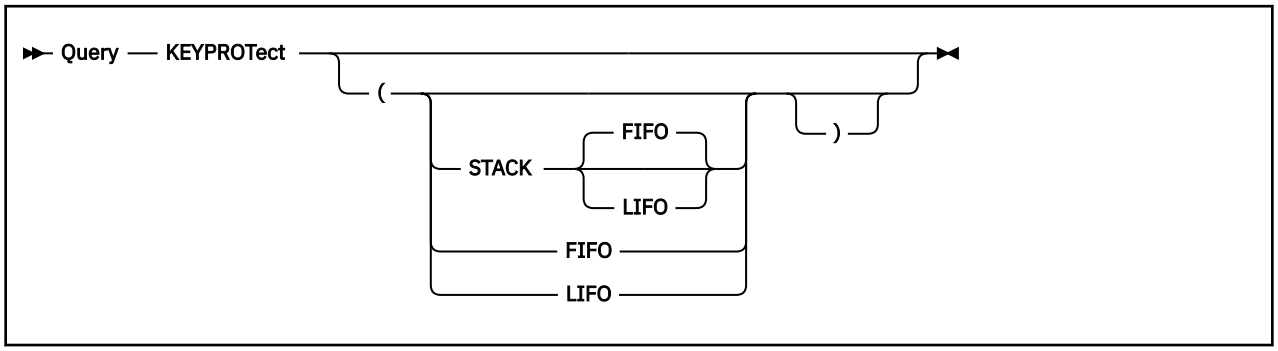
Where:

keypressed
is the attention key that was pressed. The possible values are ENTER, PAKEY *n*, PFKEY *n*, CLEAR, or UNKNOWN.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see “QUERY” on page 620.

QUERY KEYPROTECT



Authorization

General User

Purpose

Use the QUERY KEYPROTECT command to check the setting of the storage keys according to their defined values. Use the SET KEYPROTECT command to reset the user keys. For more information, see "SET KEYPROTECT" on page 953.

Responses

```
KEYPROT = ON
          OFF
```

Where:

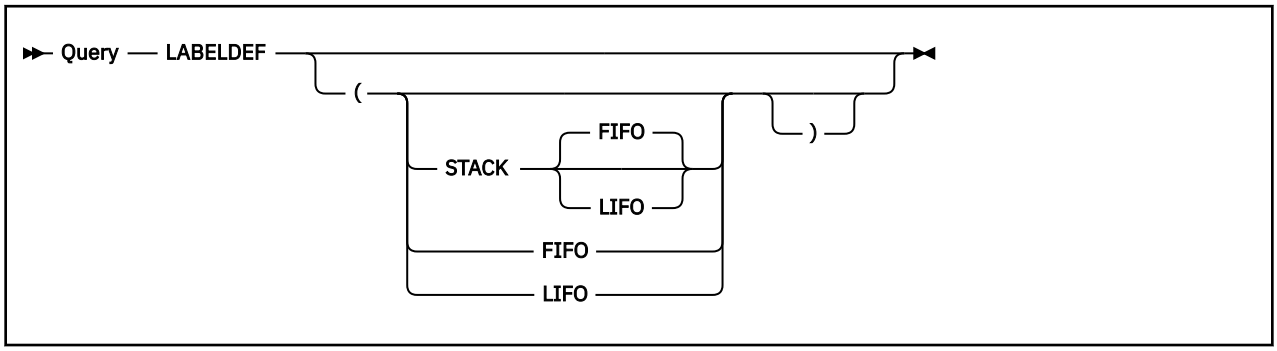
ON
indicates the storage keys were reset.

OFF
indicates the storage keys were not reset.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see "QUERY" on page 620.

QUERY LABELDEF



Authorization

General User

Purpose

Use the QUERY LABELDEF command to display all label definitions in effect.

Responses

```
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. (Exception: if the user specified the GENN and GENV parameters on the LABELDEF, the command will have automatically appended a Generation Dataset Group (GDG) number in the form: '.G*nnnn*V*nn*' to the user's 17–character HDR1 File Identifier (FID). The query will force a display of the FID data, whether it was defaulted or not, for visibility to the user. The automatic append of the GDG number may have caused left-sided truncation of the user's original FID data.) If no label definitions are in effect, the following message is displayed at the terminal:

```
No user defined LABELDEFs in effect
```

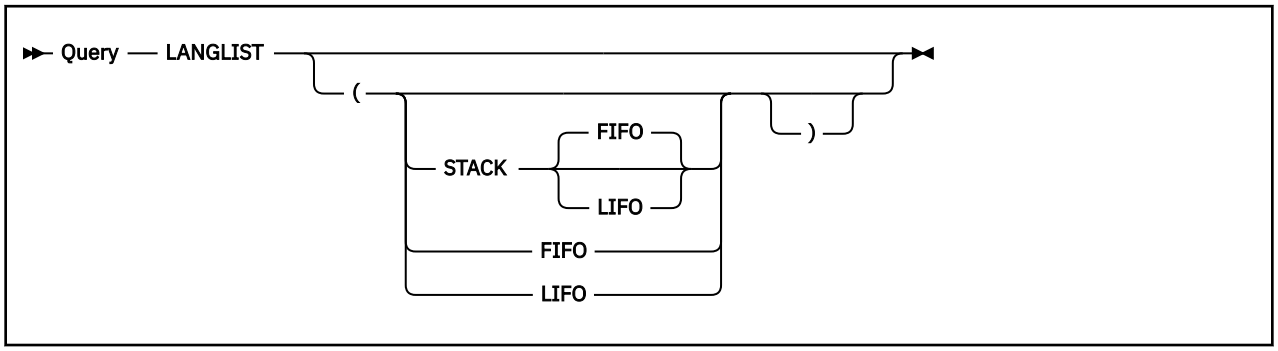
For an OS simulation user, if SCRATCH was entered at command time and the file has not been opened, all the volume IDs and SCRATCH will be displayed. If SCRATCH was entered at command time, and the file has been opened, all the volume IDs and the volume IDs of all the scratch tapes will be displayed. 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:
   .      .      .      .        .      .         .      .
   .      .      .      .        .      .         .      .
```

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see “QUERY” on page 620.

QUERY LANGLIST



Authorization

General User

Purpose

Use the QUERY LANGLIST command to display all language identifiers (*langids*) that can be set for CMS in your virtual machine, and to determine whether a certain language is valid for your virtual machine.

Responses

```
langid1
langid2
langid3
   ⋮
```

Where:

langid1
is the language identifier of the default language for CMS.

langid2
is the language identifier of the language currently active for CMS in your virtual machine. If the current, active language is not the default language, it will be displayed first among the additionally available *langid*s.

langid3...
are other valid language identifiers.

Note:

1. SYSTEM LANGUAGE S is a file that contains a list of national languages installed on your system. If this file is missing, a return code of 28 will be issued by QUERY LANGLIST.
2. The response may change if you alter the size of your virtual machine.

Sample Response

```
AMENG
UCENG
```

Where:

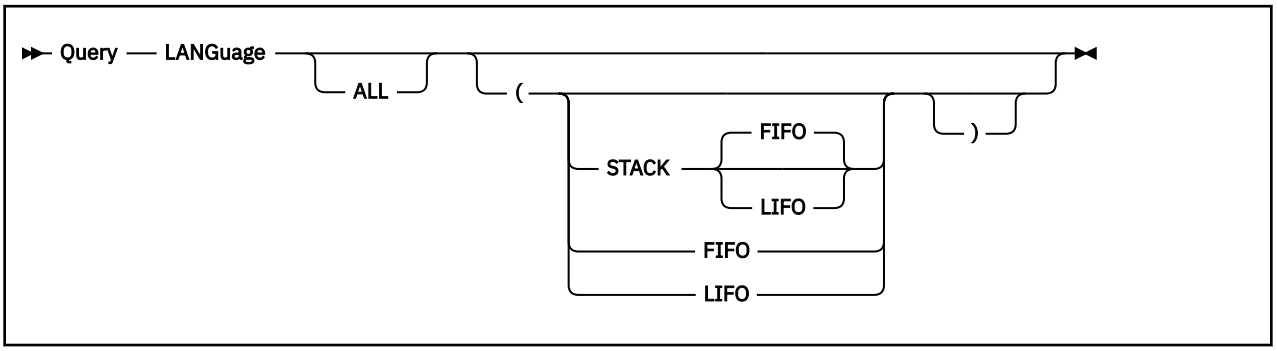
AMENG
is American English, the default language for CMS.

UCENG
is uppercase English. Either AMENG or UCENG may be the language currently active for CMS in your virtual machine.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see “QUERY” on page 620.

QUERY LANGUAGE



Authorization

General User

Purpose

Use the QUERY LANGUAGE command to display the active language identifier and information about all active application identifiers. Use the SET LANGUAGE command to change the language of your CMS session and applications. For more information, see "SET LANGUAGE" on page 954.

Operands

ALL
specifies information is to be displayed about all active application identifiers.

Responses

1. For `QUERY LANGUAGE`

```
langid
```

Where:

langid
is the language identifier of the language currently active for CMS in your virtual machine.

2. For `QUERY LANGUAGE ALL`

```
langid
applid1
USER|SYSTEM|ALL
applid2
USER|SYSTEM|ALL
⋮
```

Where:

langid
is the language identifier of the language currently active for CMS in your virtual machine.

applid1 applid2 ...
is an application identifier.

USER
indicates user repositories, command syntax tables, or command synonym tables are loaded into storage.

SYSTEM
indicates the system-provided language files for the application named are active. No user language files are active.

ALL
specifies the system-provided language files *and* user additions are active.

Example

The response to QUERY LANGUAGE ALL may be:

```
AMENG
DMS
ALL
AWG
SYSTEM
```

Where:

AMENG
specifies the language ID for American English.

DMS
specifies the application ID for CMS.

ALL
specifies you have made additions to the CMS message repository, command syntax files, or both.

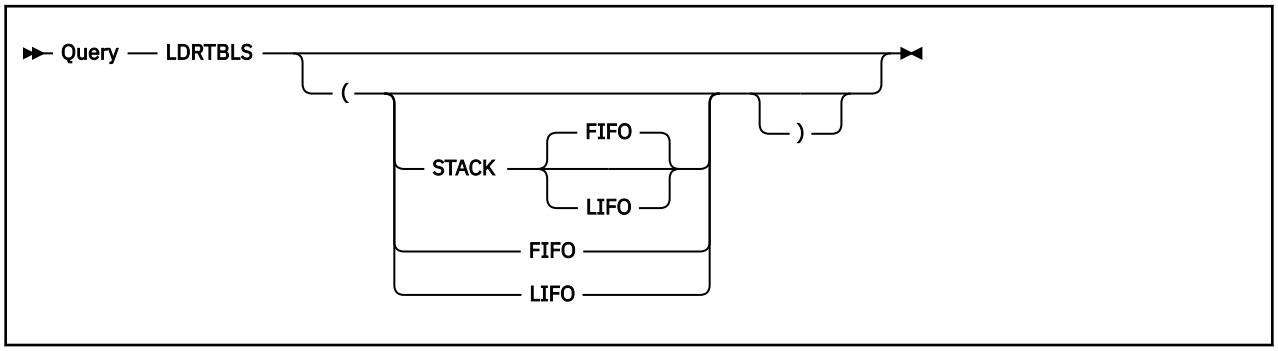
AWG
specifies the application ID for an application program.

SYSTEM
specifies you are using only the system language information for the AWG application.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see “QUERY” on page 620.

QUERY LDRTBLS



Authorization

General User

Purpose

Use the QUERY LDRTBLS command to display the number of loader tables. Use the SET LDRTBLS command to define the number of tables used. For more information, see "SET LDRTBLS" on page 958.

Responses

```
LDRTBLS = nn
```

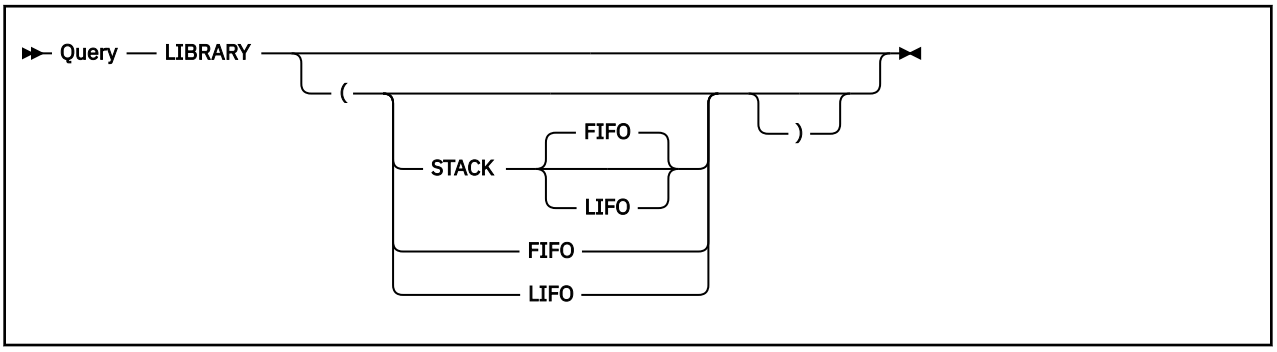
Where:

nn
is the number of loader tables.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see "QUERY" on page 620.

QUERY LIBRARY



Authorization

General User

Purpose

Use the QUERY LIBRARY command to display the names of all library files with file types of MACLIB, TXTLIB, CSLLIB, DOSLIB, and LOADLIB to be searched.

Responses

```
libtype = libname1 … libname8
             .               .
             .               .
             .               .
```

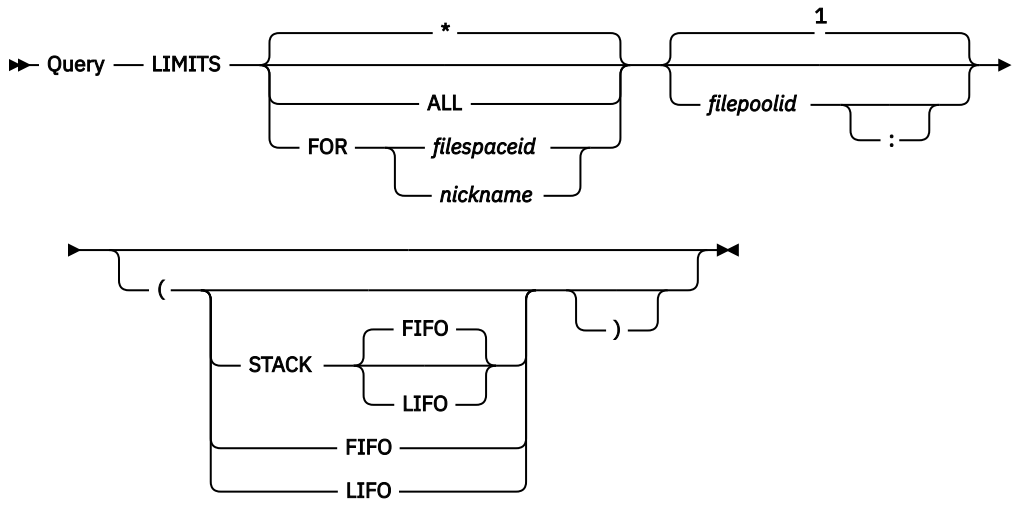
Up to eight names are displayed per line, for as many lines as necessary. If no libraries are to be searched, the response is:

```
MACLIB  = NONE
TXTLIB  = NONE
DOSLIB  = NONE
LOADLIB = NONE
CSLLIB  = NONE
```

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see “QUERY” on page 620.

QUERY LIMITS



Notes:

[1] If *filepoolid* is not specified, the default file pool ID for your virtual machine is used. If there is no default file pool ID, the command fails.

Authorization

General User for an SFS file space. Any connected user may enter the query for a byte file system.

Purpose

Use the QUERY LIMITS command to display information about your storage space limits or about a byte file system in the specified file pool.

Operands

ALL
specifies information for all file spaces should be returned.

FOR
specifies the keyword used with *filespaceid* or *nickname*.

filespaceid
identifies the user ID or the byte file system name for which you are requesting the information.

nickname
identifies a nickname that represents the name of your file space or a byte file system. Use the NAMES command to define nicknames.

*****
specifies the limit information about the user ID that entered the command is being requested.

Note: If you are a remote or batch user, the user ID queried is your APPC/VM access user ID. This is the user ID by which the file pool knows you and this verifies your authorization.

filepoolid
specifies the file pool in which the limits are being queried. If not specified, the default file pool is used. The colon is optional.

Responses

If the specified file space is enrolled in the file pool, the response is:

```
Userid   Storage Group  4K Block Limit  4K Blocks Committed  Threshold
fspaceid  stor_group       blk_limit         blks_com         thresh
```

If the file space is not enrolled, the response is the *fspaceid* and dashes in the rest of the columns.

Note: The header is generated only if output is displayed at the terminal.

Where:

fspaceid
identifies a user ID or the name of the byte file system.

stor_group
is the number of the Storage Group to which the file space has been assigned.

blk_limit
is the number of 4KB blocks available for your files or for the BFS regular files.

blks_com
is the number of 4KB blocks committed in the file space and the percentage of file space used.

thresh
For an SFS file space, this 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 for SFS file spaces.

For a byte file system, this is always 100%.

Note: If the QUERY LIMITS command is issued from an exec or assembler program for an active file pool on the specified work unit, the command will fail.

Usage Notes

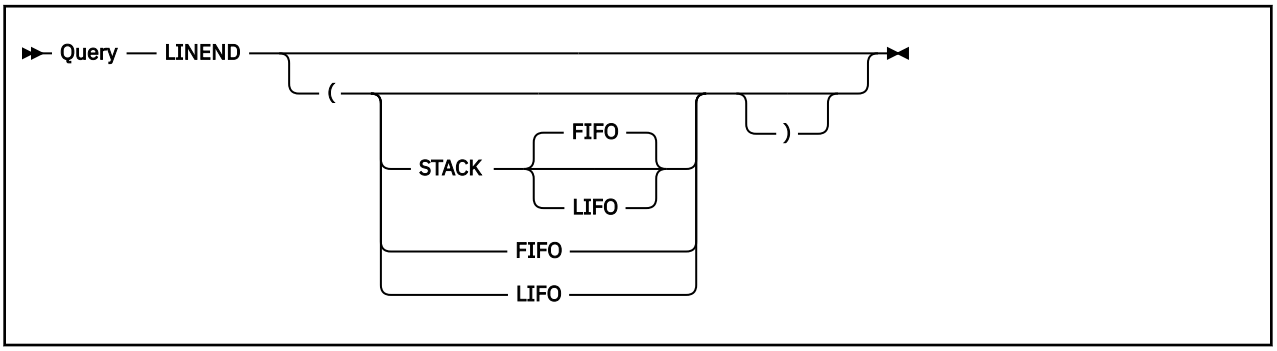
These are the valid ways to query your own limits:

- QUERY LIMITS
- QUERY LIMITS *
- QUERY LIMITS *filepoolid*
- QUERY LIMITS * *filepoolid*
- QUERY LIMITS FOR *myuserid*
- QUERY LIMITS FOR *myuserid filepoolid*

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see “QUERY” on page 620.

QUERY LINEND



Authorization

General User

Purpose

Use the QUERY LINEND command to indicate if the logical line end character is activated, and to display it. Use the SET LINEND command to define and activate the logical line end character. For more information, see "SET LINEND" on page 960.

Responses

```
LINEND   ON  char
```

or

```
LINEND   OFF char
```

Where:

ON
indicates the logical line end character is activated.

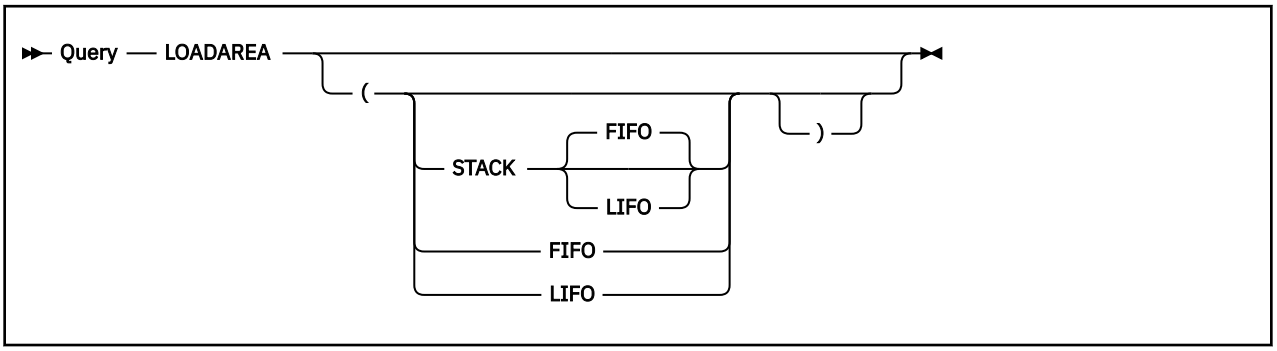
OFF
indicates the logical line end character is not activated.

char
is the logical line end character.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see "QUERY" on page 620.

QUERY LOADAREA



Authorization

General User

Purpose

Use the QUERY LOADAREA command to display the default ORIGIN for loading text files with the LOAD command. Use the SET LOADAREA command to change the default ORIGIN. For more information, see "SET LOADAREA" on page 961.

Responses

```
LOADAREA = 20000
```

or

```
LOADAREA = RESPECT
```

Where:

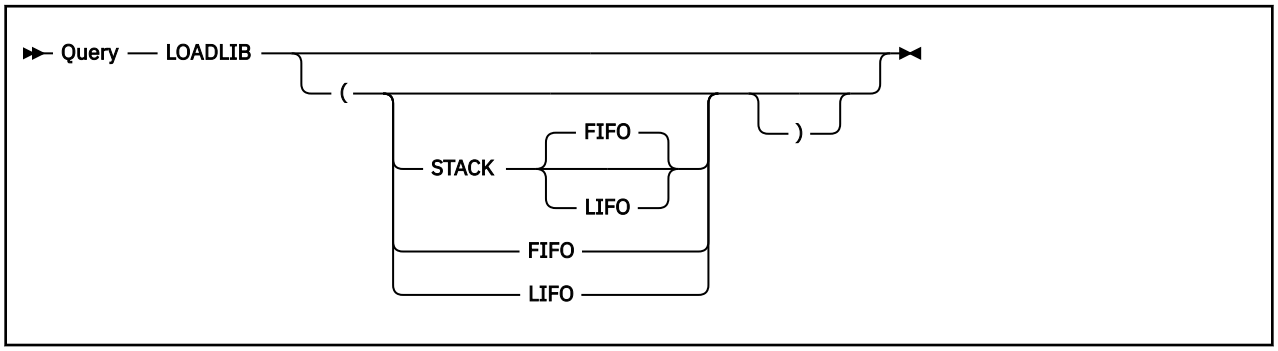
20000
indicates the LOAD command will start loading at X'20000' if the ORIGIN or RMODE option is not specified on the LOAD command.

RESPECT
indicates that the LOAD command will respect the RMODE during text file ESD processing to determine where the loaded program should reside when the RMODE or ORIGIN option is not specified on the LOAD command.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see "QUERY" on page 620.

QUERY LOADLIB



Authorization

General User

Purpose

Use the QUERY LOADLIB command to display the names of all files with a file type of LOADLIB to be searched for load modules (that is, all LOADLIBs specified on the last GLOBAL LOADLIB command, if any).

Responses

```
LOADLIB = libname1 … libname8
   .          .               .
   .          .               .
   .          .               .
```

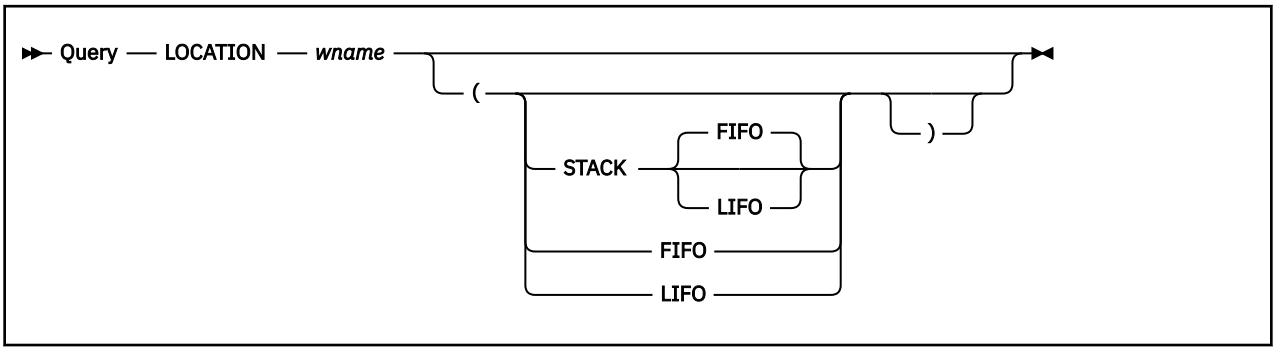
Up to eight names are displayed per line, for as many lines as necessary. If no LOADLIBs are to be searched, the following message is displayed at the terminal:

```
LOADLIB = NONE
```

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see “QUERY” on page 620.

QUERY LOCATION



Authorization

General User

Purpose

Use the QUERY LOCATION command to display whether the location indicator is displayed in the specified window when the data in the virtual screen exceeds the size of the window. Use the SET LOCATION command to turn the display of the location indicator on and off. For more information, see "SET LOCATION" on page 963.

Operands

wname
is the name of the window which may display the location indicator.

Responses

```
LOCATION wname     ON
```

or

```
LOCATION wname     OFF
```

Where:

wname
is the name of the specified window.

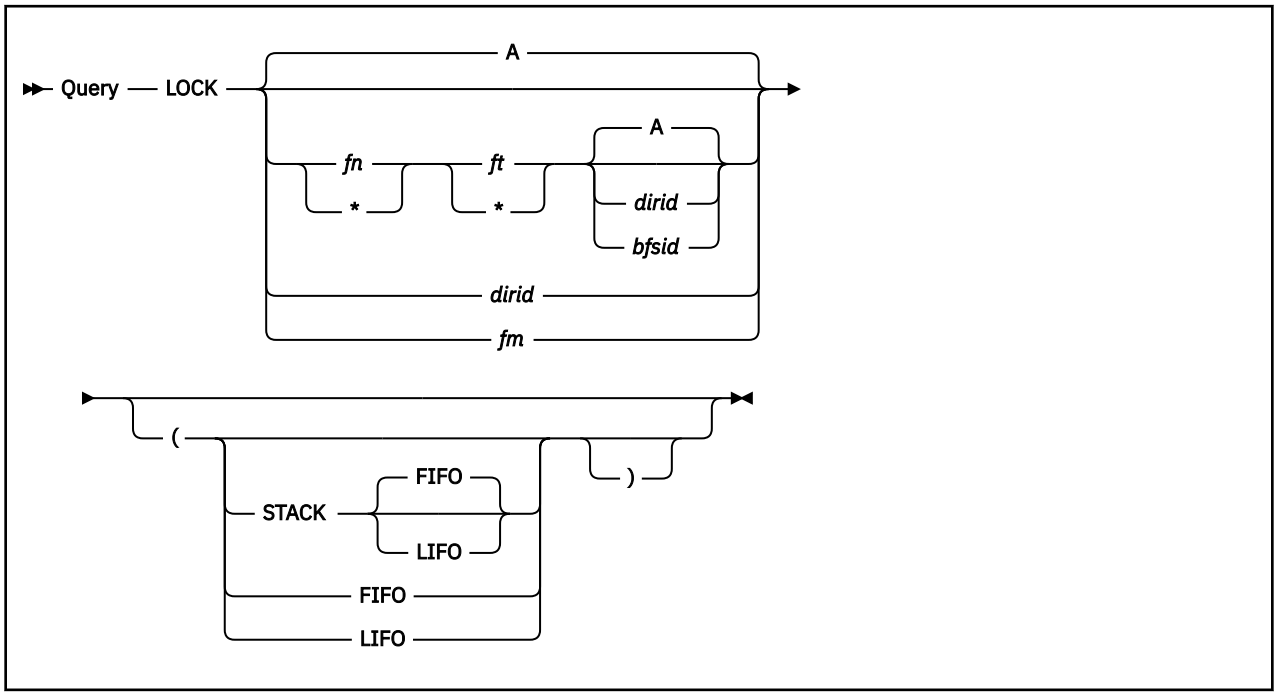
ON
indicates the location indicator is displayed when there is data to be viewed outside the window.

OFF
indicates the location indicator is not displayed.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see "QUERY" on page 620.

QUERY LOCK



Authorization

General User

Purpose

Use the QUERY LOCK command to display lock information on a Shared File System (SFS) directory, a file in a directory, or on a BFS regular file. Files and directories are locked using the CREATE LOCK command and unlocked with the DELETE LOCK command.

Operands

fn
is the file name of the file whose lock information is to be displayed.

ft
is the file type of the file whose lock information is to be displayed.

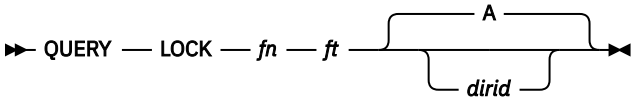
fm
is the file mode of the file whose lock information is to be displayed.

dirid
is the identifier of the directory whose lock information is to be displayed, or which contains the file whose lock information is to be displayed. If not specified, the directory accessed as A is used. You can also use a file mode for the *dirid* if the directory is already accessed. For more information on *dirid*, see "Naming Shared File System (SFS) Directories" on page 6.

bfsid
is the name of the byte file system (BFS) in which the file or files being queried reside.

For a BFS file, *fn* and *ft* are system-generated values that together uniquely identify the file within the BFS.

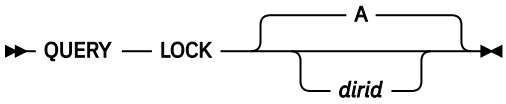
Responses



1.

```
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.



2.

```
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 parent 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 having the file control (FILECONTROL) attribute

DIRC
directory having the directory control (DIRCONTROL) attribute

ALIAS*
alias pointing to a base file in DFSMS/VM migrated status

BASE*
base file in DFSMS/VM migrated status

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 the lock owner reads or modifies

EXCLUSIVE
only the lock owner can 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 the lock owner's CMS session with the file pool ends. For more information, see "CREATE LOCK" on page 104.

If no locks are in effect, and the STACK, LIFO, or FIFO option was specified, the return code is set to 6, indicating no data was stacked.

Response Messages

- If there are no locks held on the file or directory, the response is:

```
No locks are held on fn ft dirid
```

or

```
No locks are held on directory dirid
```

Usage Notes

1. To query the locks on a file, specify *fn ft* and the *dirid* containing the file. To query locks on a directory, specify only the *dirid*. You must have authority for the file or directory you query.
2. Special characters (* or %) can be used for *fn ft* to query a set of files, providing you have appropriate authority for the directory specified by *dirid*. For FILECONTROL directories, you need read or write authority on the directory. For DIRCONTROL directories, you need DIRREAD or DIRWRITE authority on the directory.

   The locked files and directories which match your specification, and on which you have read or write authority, are displayed. For more information on *dirid*, see "Naming Shared File System (SFS) Directories" on page 6. For more information on using special characters to do pattern matching, see "Using Pattern Matching to Specify Sets of Files" on page 14.

   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 locks on

   ```
   PHILLIPS NOTEBOOK
   PHILLIPSNOTES
   ```

   where PHILLIPS NOTEBOOK is a file and PHILLIPSNOTES is a subdirectory. Issuing,

   ```
   query lock phil* n* a
   ```

   would find both PHILLIPS NOTEBOOK and PHILLIPSNOTES because PHILLIPSNOTES contains more than eight characters and is matched as if it had a file name of PHILLIPS and a file type of NOTES. Issuing,

   ```
   query lock phillipsnotes * a
   ```

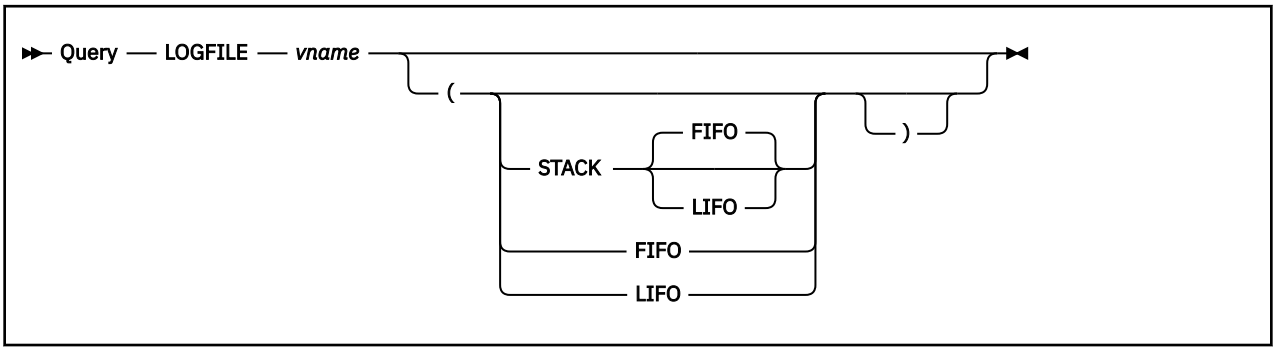
   would also find both PHILLIPS NOTEBOOK and PHILLIPSNOTES because only the first eight characters are used as the file name. Therefore, any file or alias with the name of PHILLIPS, or any subdirectory with PHILLIPS as the first eight characters in its name would be listed if it has a lock.

3. If the QUERY LOCK command does not resolve the type of lock that exists, there may be an outstanding disable lock in the object's directory hierarchy. For possible disabled lock on a file space or its owning storage group, see “QUERY FILEPOOL DISABLE” on page 694.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see “QUERY” on page 620.

QUERY LOGFILE



Authorization

General User

Purpose

Use the QUERY LOGFILE command to display whether or not a log file is updated with data written to the virtual screen. Use the SET LOGFILE command to control updating of the log file. For more information, see "SET LOGFILE" on page 965.

Operands

vname
is the name of the virtual screen whose log file may be being updated.

Responses

```
LOGFILE vname    ON  fn ft fm
```

or

```
LOGFILE vname    OFF fn ft fm
```

Where:

ON
indicates a log file is updated with data written to the virtual screen.

OFF
indicates 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.

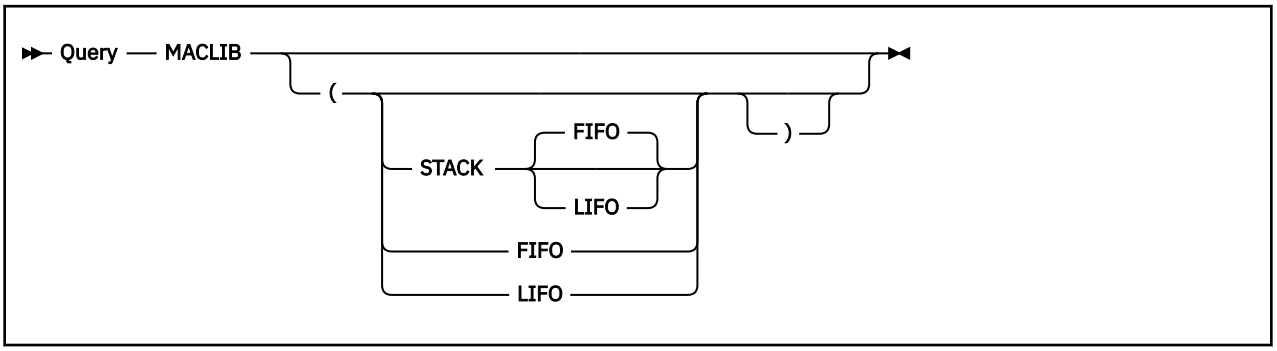
ft
is the file type of the log file.

fm
is the file mode of the log file.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see “QUERY” on page 620.

QUERY MACLIB



Authorization

General User

Purpose

Use the QUERY MACLIB command to display the names of all files, with a file type of MACLIB, to be searched for macro definitions (that is, all MACLIBs specified on the last GLOBAL MACLIB command, if any).

Responses

```
MACLIB = libname1 … libname8
  .          .                .
  .          .                .
  .          .                .
```

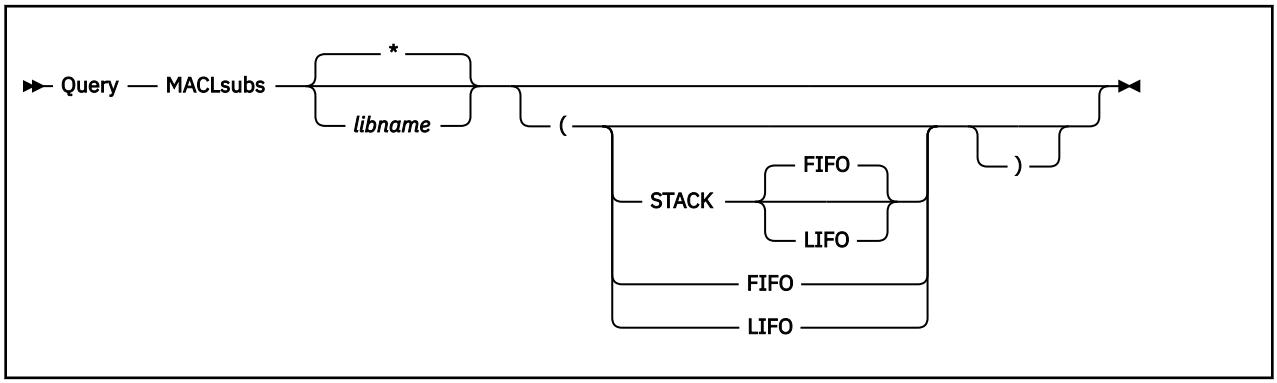
Up to eight names are displayed per line, for as many lines as necessary. If no macro libraries are to be searched for macro definitions, the response is:

```
MACLIB = NONE
```

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see “QUERY” on page 620.

QUERY MACLSUBS



Authorization

General User

Purpose

Use the QUERY MACLSUBS command to display the macro library substitutions defined by the SET MACLSUBS command.

Operands

*****
specifies all macro libraries are queried. All macro library substitutions that have been set are displayed. This is the default.

libname
is the name of the macro library to be queried. The macro library substitutions that have been set for this library name are displayed.

Responses

```
MACLSUBS libname libsub1 libsub2...(option
```

or

```
DMSSML2508W No library substitutions in effect [for libname] [RC=4]
```

Where:

libname
is the macro library name to be replaced.

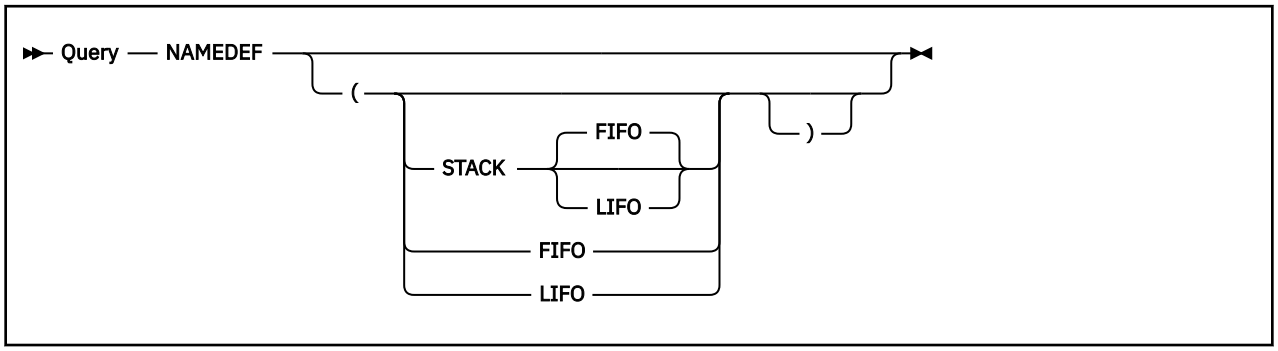
libsub1 libsub2...
are the substitute macro library names. Up to six substitute library names are displayed per line.

option
is INFORM or NOINFORM to specify whether an informational message (DMSSML2152I) is issued when the GLOBAL command is replacing the macro libraries.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see “QUERY” on page 620.

QUERY NAMEDEF



Authorization

General User

Purpose

Use the QUERY NAMEDEF command to display the current name definitions that were established with the CREATE NAMEDEF command. For more information, see "CREATE NAMEDEF" on page 108.

Responses

```
Namedef      Filename    Filetype    Filemode/Directory Name
namedef      -           -           dirname
namedef      fn          ft          -
namedef      -           -           fm
namedef      -           -           bfsid
  .          .           .           .
  .          .           .           .
```

Note: The header is generated only if output is displayed at the terminal.

Where:

namedef
is a temporary name associated with a:

- File name and file type
- Directory name
- File mode letter

fn
is the file name associated with the name definition. If a dash is displayed, it means a directory name or file mode letter is associated with the name definition.

ft
is the file type associated with the name definition. If a dash is displayed, it means a directory name or file mode letter is associated with the name definition.

dirname
is the complete directory name associated with the name definition. If a dash is displayed, it means only a file name and file type are associated with the name definition.

fm
is the file mode letter associated with the name definition. If a dash is displayed, it means only the file name and file type are associated with the name definition.

bfsid
is the name of the byte file system associated with the name definition. If a dash is displayed, it means only a file name and file type are associated with the name definition.

If you have not specified any name definitions, you will get the following response:

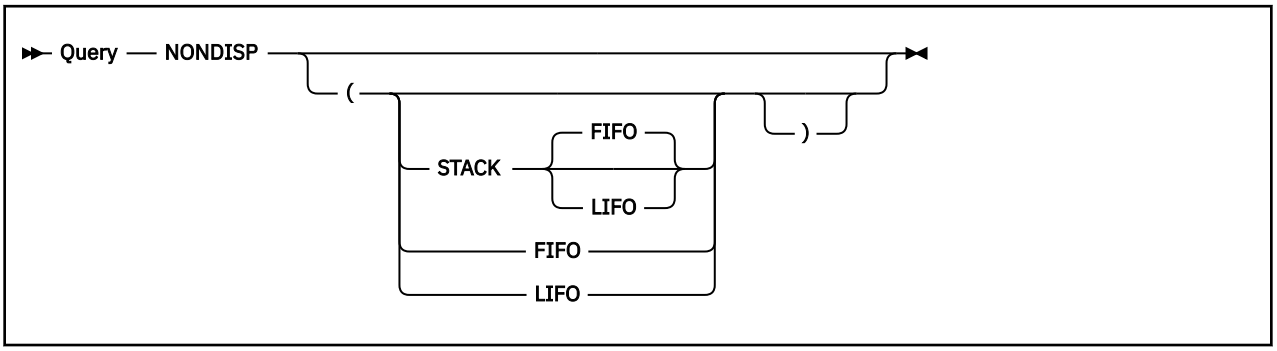
```
No user defined NAMEDEF in effect
```

If no name definitions are in effect and the STACK, LIFO, or FIFO option was specified, the return code is set to 6, indicating no data was stacked.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see “QUERY” on page 620.

QUERY NONDISP



Authorization

General User

Purpose

Use the QUERY NONDISP command to show the character set to be displayed in place of nondisplayable characters. Use the SET NONDISP command to specify the character. For more information, see "SET NONDISP" on page 969.

Responses

```
NONDISP     char
```

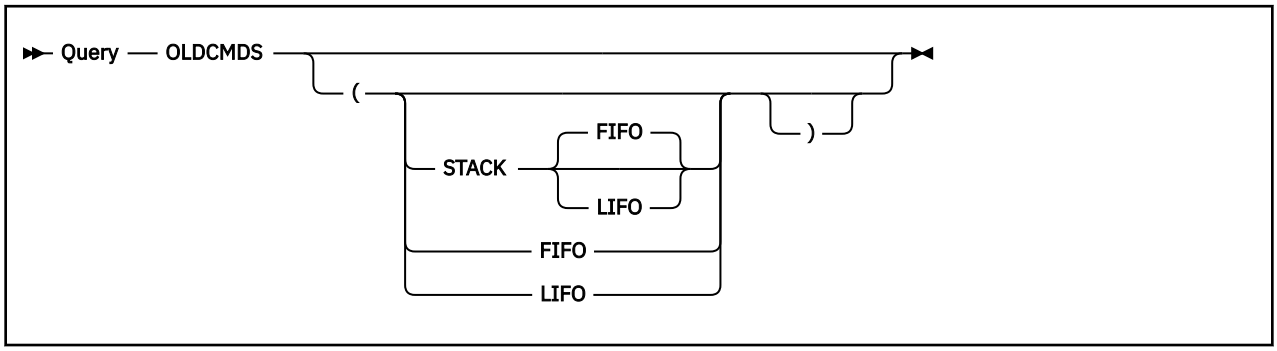
Where:

char
is the character displayed in place of nondisplayable characters.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see "QUERY" on page 620.

QUERY OLDCMDS



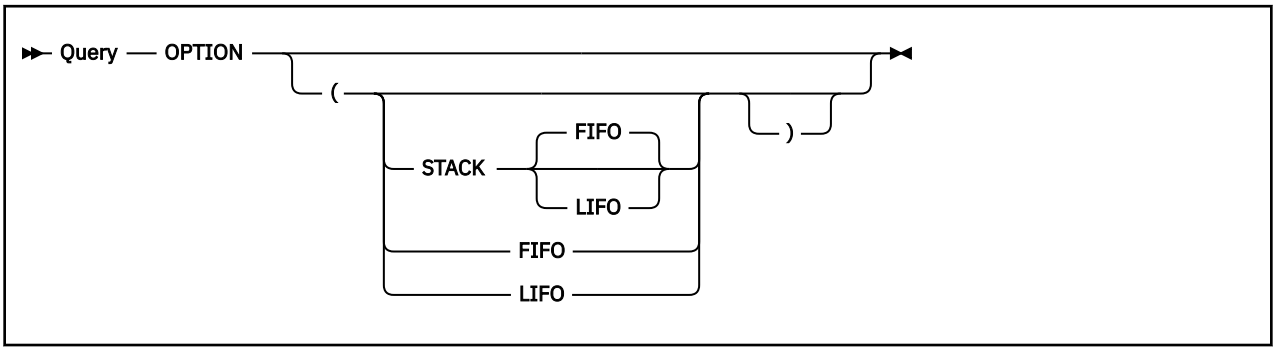
Authorization

General User

Purpose

Use the QUERY OLDCMDS command to display whether the original formats of the Session Services commands were available for invocation. If they no longer are available, QUERY OLDCMDS will always return OFF. This command is maintained for compatibility purposes only.

QUERY OPTION



Authorization

General User

Purpose

Use the QUERY OPTION command to display the compiler options currently in effect for the DOS/VS COBOL compiler and the RPG II compiler in CMS/DOS.

Responses

The possible options that can be in effect, in the order they will be listed, are:

```
OPTION = deck list listx sym xref errs charset dump term
```

Where:

deck
specifies a return value of DECK or NODECK. 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.

list
specifies a return value of LIST or NOLIST. LIST writes the output listing of the source module on the SYSLST device.

listx
specifies a return value of LISTX or NOLISTX. LISTX produces a procedure division map on the SYSLST device.

sym
specifies a value of SYM or NOSYM. SYM prints a Data Division map on SYSLST.

xref
specifies a value of XREF or NOXREF. XREF writes the output symbolic cross-reference list on SYSLST.

errs
specifies a value of ERRS or NOERRS. ERRS writes an output listing of all errors in the source program on SYSLST.

charset
specifies the character set in effect. A 48C will be returned for the 48-character set, and 60C will be returned for the 60-character set.

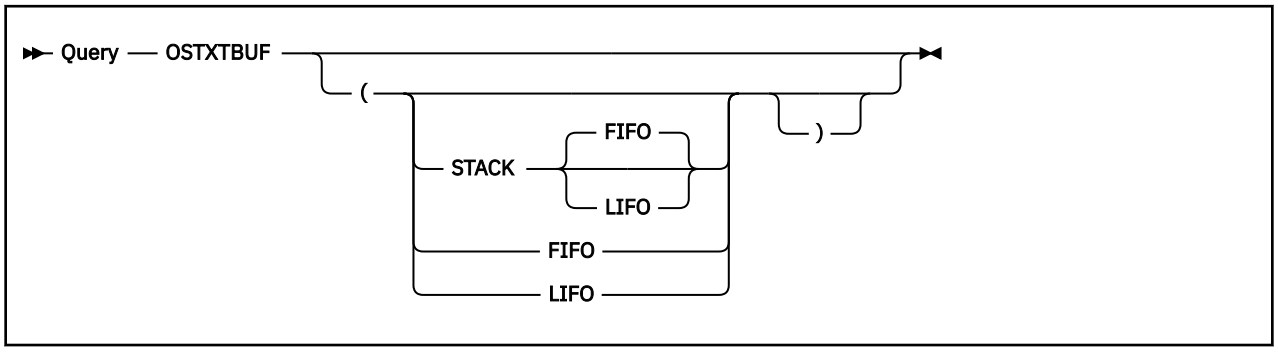
dump
specifies a value of DUMP or NODUMP. DUMP dumps the registers and the virtual partition on the virtual SYSLST device in the case of abnormal program end.

term
specifies a value of TERM or NOTERM. TERM displays all compiler messages to the user's terminal.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see "QUERY" on page 620.

QUERY OSTXTBUF



Authorization

General User

Purpose

Use the QUERY OSTXTBUF command to display the current setting of the SET OSTXTBUF command. Use the SET OSTXTBUF command to control the size of the storage buffer reserved before OS loading text files. For more information, see "SET OSTXTBUF" on page 974.

Responses

```
OSTXTBUF = hexnum
```

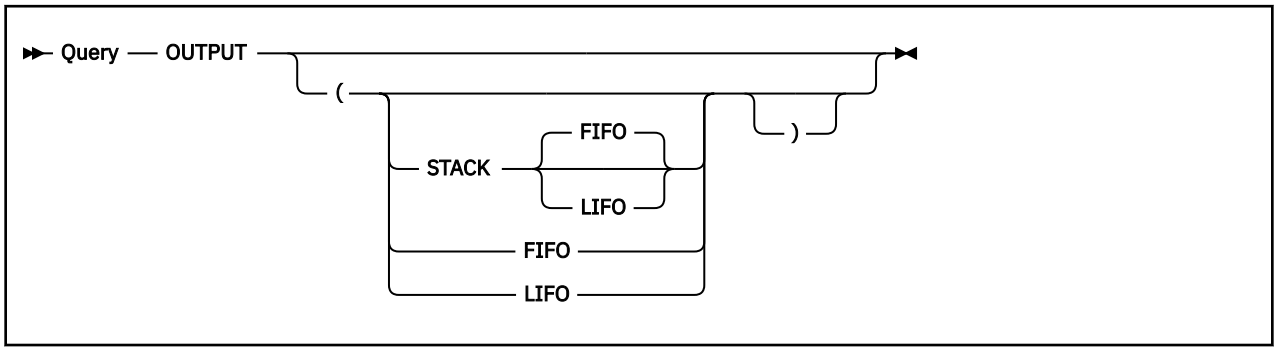
Where:

hexnum
indicates the hexadecimal number of bytes in storage that will be added to the current load location for OS loaded text files. The default value is 50000.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see "QUERY" on page 620.

QUERY OUTPUT



Authorization

General User

Purpose

Use the QUERY OUTPUT command to display the contents of any output translate table in effect. Use the SET OUTPUT command to set up output translation tables. For more information, see "SET OUTPUT" on page 976.

Responses

```
OUTPUT = xx1  a1
           .   .
           .   .
           .   .
         xxn  an
```

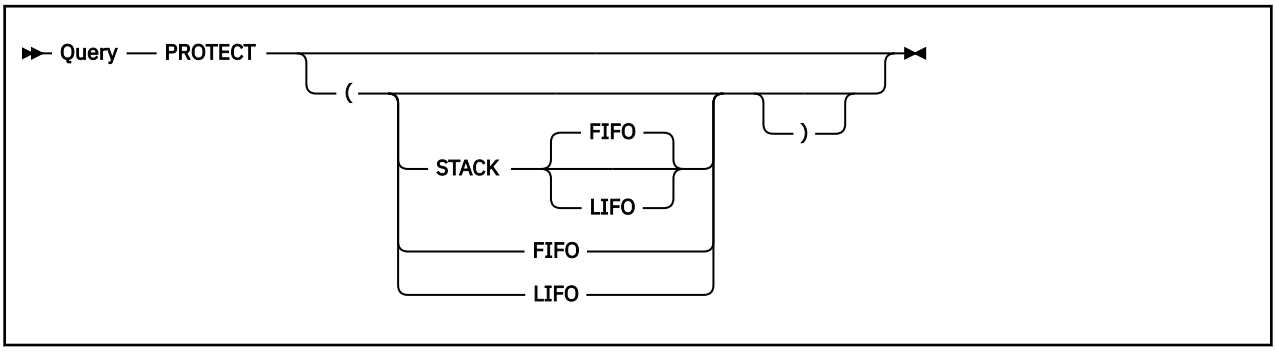
If you do not have an output translate table defined, the response is:

```
No user defined output translate table in use
```

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see "QUERY" on page 620.

QUERY PROTECT



Authorization

General User

Purpose

Use the QUERY PROTECT command to display the status of CMS nucleus protection. Use the SET PROTECT command to set nucleus protection. For more information, see "SET PROTECT" on page 977.

Responses

```
PROTECT  = ON
```

or

```
PROTECT  = OFF
```

Where:

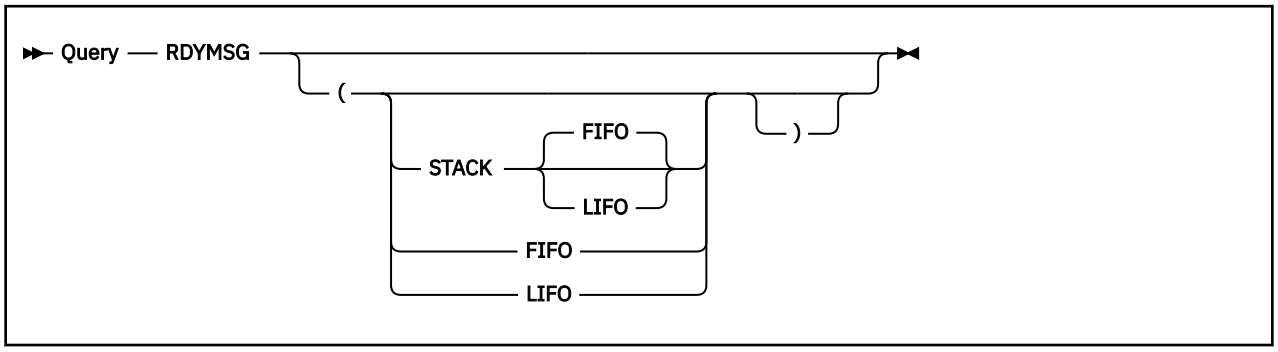
ON
means CMS nucleus protection is in effect.

OFF
means CMS nucleus protection is not in effect.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see "QUERY" on page 620.

QUERY RDYMSG



Authorization

General User

Purpose

Use the QUERY RDYMSG command to display the format of the CMS ready message in effect. Use the SET RDYMSG command to change the format of the message. For more information, see "SET RDYMSG" on page 978.

Responses

```
RDYMSG = LMSG SRC
```

or

```
RDYMSG = LMSG LRC
```

or

```
RDYMSG = SMSG SRC
```

or

```
RDYMSG = SMSG LRC
```

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;
```

LRC
is an expanded return code that will display up to ten digits, plus a minus sign:

```
Ready; (1234567890)
Ready; (-1234567890)
```

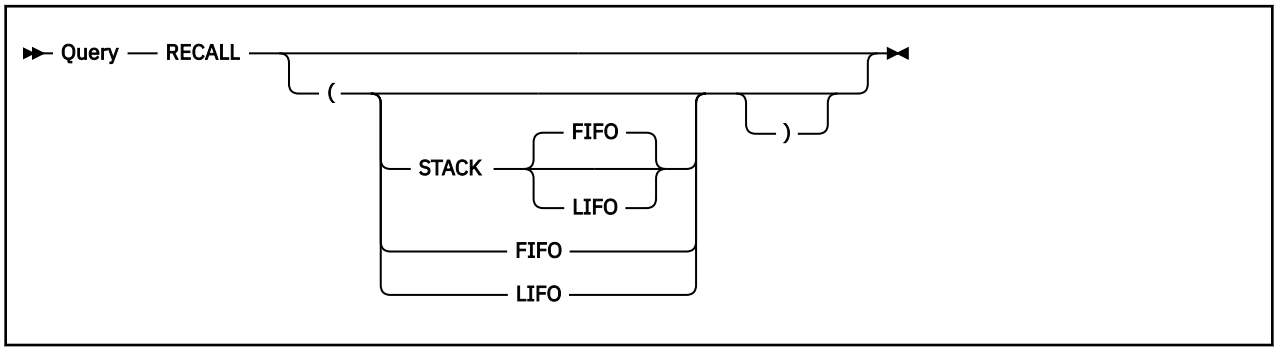
SRC
is the standard five digit return code. A return code greater than five digits will be truncated:

```
Ready; (67890)
Ready; (-7890)
```

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see “QUERY” on page 620.

QUERY RECALL



Authorization

General User

Purpose

Use the QUERY RECALL command to display whether SFS files that have been placed in DFSMS/VM migrated status (that is, have had their data moved into the DFSMS/VM storage repository) are to be implicitly recalled when file data is referenced. Change this setting with the SET RECALL command. For more information, see "SET RECALL" on page 980.

Responses

```
RECALL ON
```

or

```
RECALL OFF
```

Where:

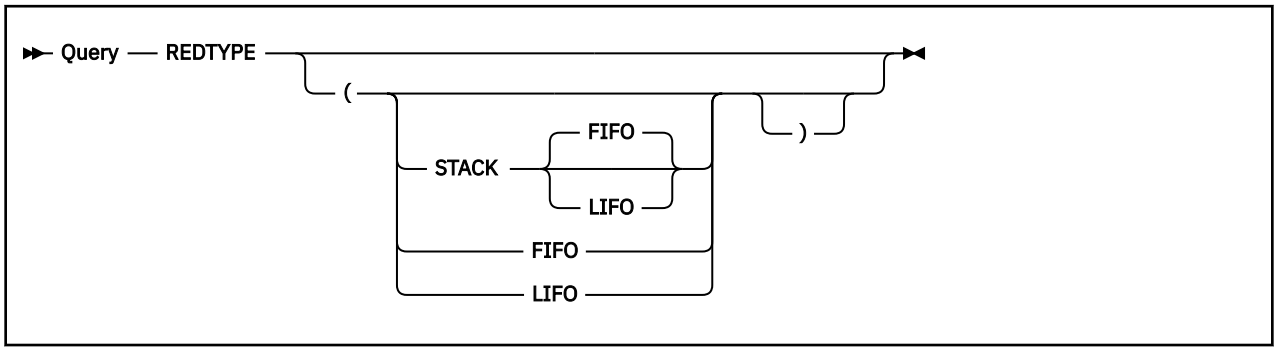
ON
indicates files that have been placed in DFSMS/VM migrated status will have their data implicitly recalled from the DFSMS/VM storage repository when they are required as input.

OFF
indicates referring to files in DFSMS/VM migrated status will not cause their data to be implicitly recalled. You will get error message if you try to refer to the file data when RECALL is OFF. You must issue the DFSMS RECALL command to recall such a file prior to opening it for input or update.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see "QUERY" on page 620.

QUERY REDTYPE



Authorization

General User

Purpose

Use the QUERY REDTYPE command to display the status of the REDTYPE indicator. Use the SET REDTYPE command to change the REDTYPE setting. For more information, see “SET REDTYPE” on page 982.

Responses

```
REDTYPE  = ON
```

or

```
REDTYPE  = OFF
```

Where:

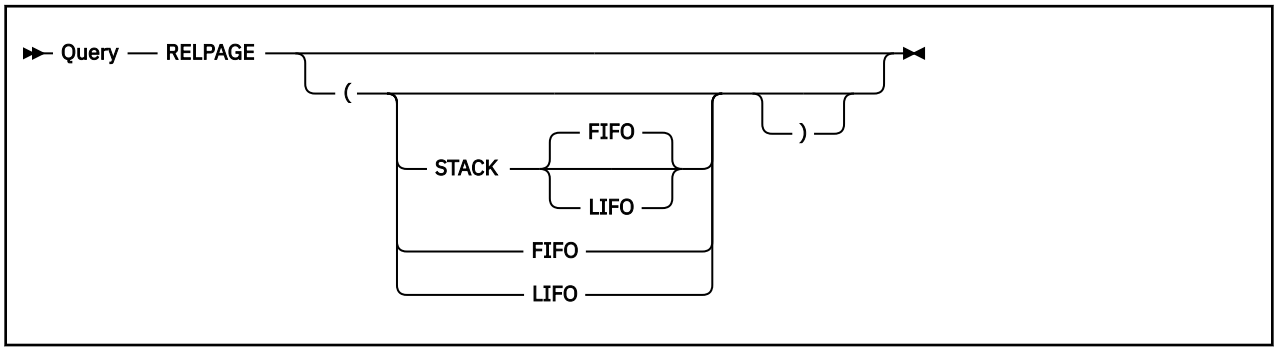
ON
types CMS error messages in red, for certain terminals equipped with the appropriate terminal feature and a two-color ribbon.

OFF
does not type CMS error messages in red.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see “QUERY” on page 620.

QUERY RELPAGE



Authorization

General User

Purpose

Use the QUERY RELPAGE command to display whether pages of storage are to be released or retained after certain commands complete execution. Use the SET RELPAGE command to release or retain pages of storage. For more information, see “SET RELPAGE” on page 983.

Responses

```
RELPAGE  = ON
```

or

```
RELPAGE  = OFF
```

Where:

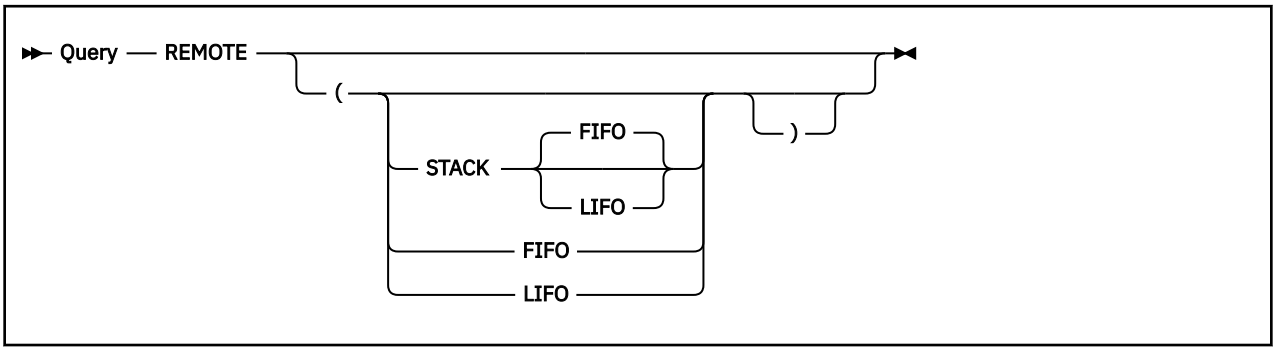
ON
releases pages.

OFF
retains pages.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see “QUERY” on page 620.

QUERY REMOTE



Authorization

General User

Purpose

Use the QUERY REMOTE command to display the manner in which data transmission is handled. Use the SET REMOTE command to change the data transmission setting. For more information, see "SET REMOTE" on page 984.

Responses

```
REMOTE   ON
```

or

```
REMOTE   OFF
```

Where:

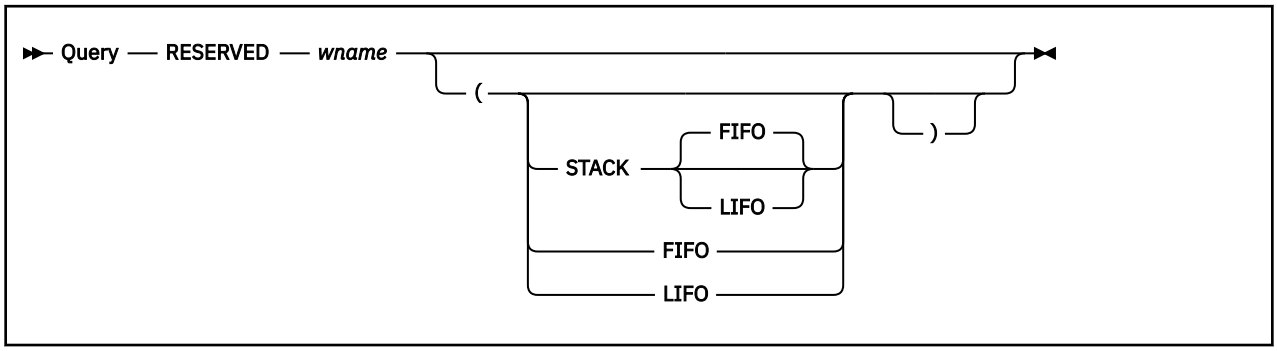
ON
specifies data is compressed by removing nulls and combining data when five or more of the same characters occur consecutively in a data stream.

OFF
specifies the data stream is not compressed. Data is transmitted with no minimization.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see "QUERY" on page 620.

QUERY RESERVED



Authorization

General User

Purpose

Use the QUERY RESERVED command to display the maximum number of reserved lines on the top and bottom of the specified window. Use the SET RESERVED command to specify the reserved lines. For more information, see "SET RESERVED" on page 985.

Operands

wname
is the name of the window whose number of reserved lines is to be displayed.

Responses

```
RESERVED wname rtop rbot
               *    *
```

Where:

wname
is the name of the window.

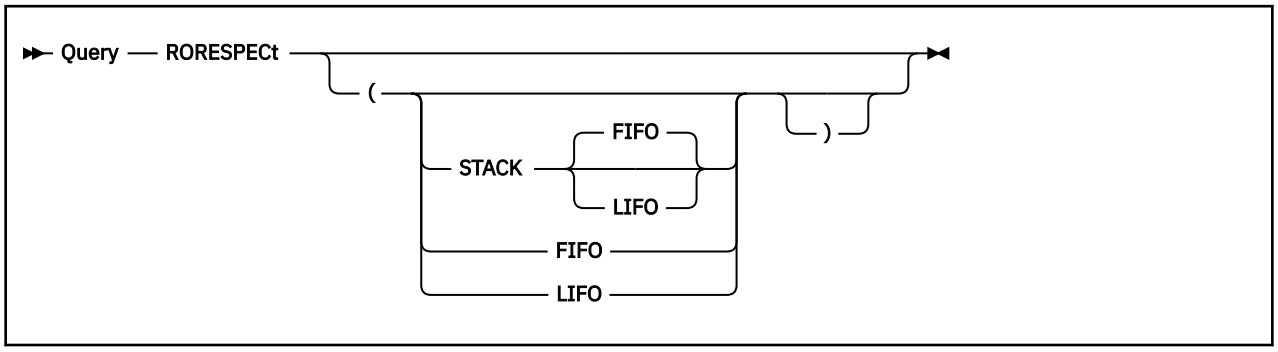
rtop
is the maximum number of top reserved lines that can be displayed.

rbot
is the maximum number of bottom reserved lines that can be displayed.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see "QUERY" on page 620.

QUERY RORESPECT



Authorization

General User

Purpose

Use the QUERY RORESPECT command to display the current RORESPECT setting. This setting controls whether XEDIT and COPYFILE respect a read-only access of a SFS file control directory on updates. Use SET RORESPECT to change the current setting. For more information, see "SET RORESPECT" on page 987.

Responses

```
RORESPECT ON
```

or

```
RORESPECT OFF
```

Where:

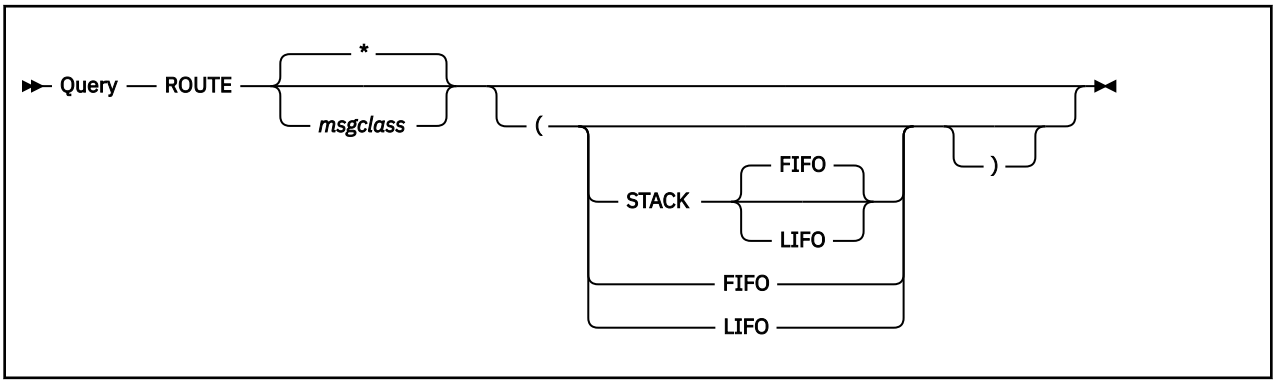
ON
: specifies attempts to update files (using XEDIT and COPYFILE) in SFS file control directories accessed read-only will not be based on authorization, but with respect to the access mode.

OFF
: specifies attempts to update files (using XEDIT and COPYFILE) in SFS file control directories will be based on authorization without respect to the access mode.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see "QUERY" on page 620.

QUERY ROUTE



Authorization

General User

Purpose

Use the QUERY ROUTE command to display the routing of message classes when SET FULLSCREEN is set to ON or SUSPEND.

Operands

msgclass
is the message class about which information is to be displayed.

*****
displays the routing of all message classes. This is the default.

Responses

For

```
QUERY ROUTE msgclass
QUERY ROUTE *
```

One line of information for each message class is displayed.

```
msgclass  TO  vname  (alarm  notify
```

Where:

msgclass
is the message class which is directed to the virtual screen. It may be:

- CMS
- CP
- MESSAGE
- WARNING
- SCIF
- NETWORK

vname
is the virtual screen receiving the output.

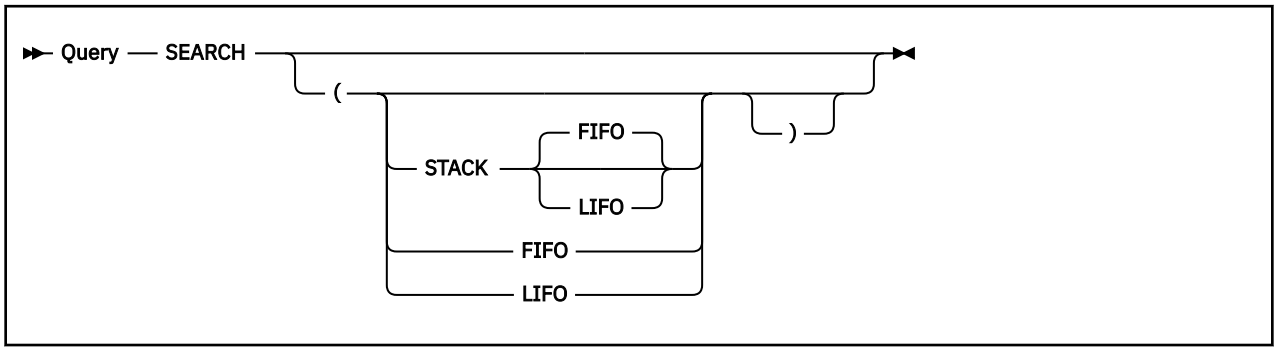
alarm
is ALARM if the alarm is sounded when a message is received and NOALARM if the alarm does not sound when a message is received.

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.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see "QUERY" on page 620.

QUERY SEARCH



Authorization

General User

Purpose

Use the QUERY SEARCH command to display the search order of accessed disks and Shared File System (SFS) Directories.

Responses

```
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 number for a disk or "DIR" for an SFS directory. If the directory has the directory control (DIRCONTROL) attribute, ‘DIRC’ is displayed instead of ‘DIR’.

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 from 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.

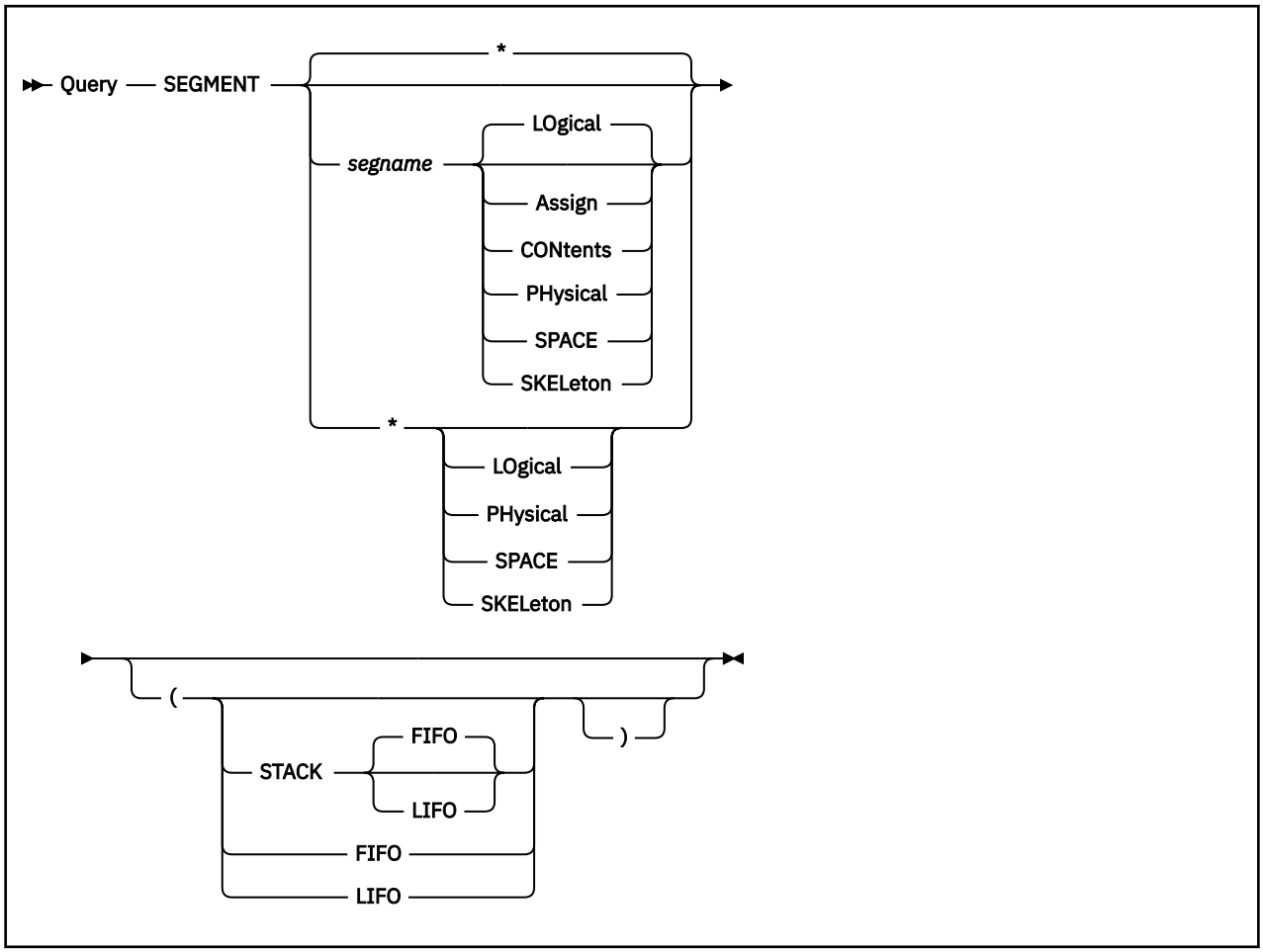
stat
indicates whether the status of the disk or SFS directory is read/write (R/W) or read/only (R/O).

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.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see “QUERY” on page 620.

QUERY SEGMENT



Authorization

General User

Purpose

Use the QUERY SEGMENT command to display information about saved segments and segment storage spaces reserved and loaded by the SEGMENT command For more information, see “SEGMENT” on page 875. For more information on the SEGMENT macro see, *z/VM: CMS Macros and Functions Reference*.

Operands

*****
indicates information about all saved segments, or all saved segments of the specified type, or all segment storage spaces is to be returned. The default is to return information about all saved segments.

segname
is the name of the saved segment about which information is to be displayed.

LOgical
specifies logical saved segment information is to be returned. If *segname* is specified, this is the default.

Assign
returns the name of the physical saved segment the specified logical saved segment is assigned to.

CONtents
returns a list of the objects in the specified logical saved segment.

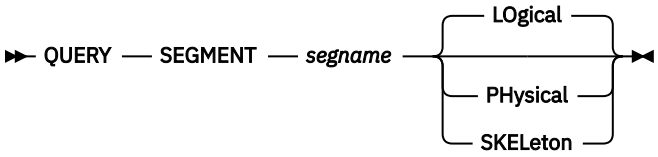
PHysical
specifies physical saved segment information is to be returned.

SPACE
specifies segment storage space information is to be returned.

SKELeton
specifies information about skeleton segments (reserved with the SKELeton option) is to be returned.

Responses

1. For




```
Space   Name     Location  Length   Loaded  Attribute
space   segname  location  length   YES/NO  SYSTEM/USER
```

Where:

space
is the name of the storage space that contains or is reserved for the saved segment:

- If *segname* is a logical saved segment, *space* is the physical saved segment.
- If *segname* is a physical saved segment that is a DCSS, *space* is the DCSS.
- If *segname* is a physical saved segment that is a member of a CP segment space, *space* is the CP segment space.
- If *segname* is a skeleton segment (reserved with the SKELeton option), *space* is the same as *segname*.

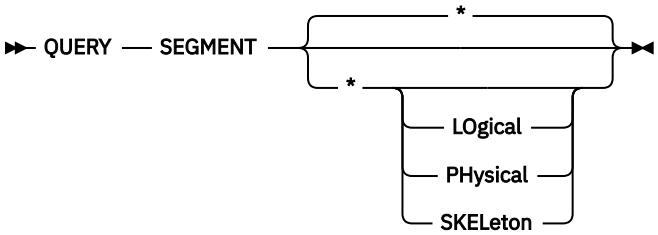
segname
is the name of the saved segment.

location
is the starting address of saved segment.

length
is the length of the saved segment.

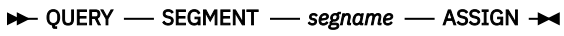
The loaded column indicates whether the saved segment is attached to the virtual machine. If SYSTEM is returned in the Attribute column, it means the segment storage space will survive abend processing. If USER is returned, it indicates the segment storage space will not survive abend processing.

2. For



The response is the same as for QUERY SEGMENT *segname* [LOgical|PHysical]; one line is returned for each logical, physical or skeleton saved segment.

3. For



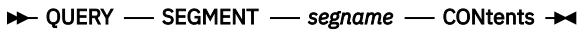

```
Lsegname  Assignment
lsegname  psegname
```

Where:

lsegname
is the name of the specified logical saved segment.

psegname
is the name of the physical saved segment the logical saved segment is assigned to.

4. For




```
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 saved segment.

length
is the length of the object in the saved segment.

name
is the name of the object in the saved segment.

Example

Enter:

```
query segment myseg contents
```

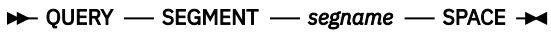
This will list the contents of the logical saved segment named MYSEG:

```
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
```

5. For




```
Space   Name    Location  Length   Loaded  Attribute
space   -       location  length   YES/NO  -
```

Where:

space
is the name of the storage space that contains or is reserved for the saved segment:

- If *segname* is a physical saved segment that is a DCSS, or if *segname* is a logical saved segment defined in a physical saved segment that is a DCSS, *space* is the DCSS.
- If *segname* is a physical saved segment that is a member of a CP segment space, or if *segname* is a logical saved segment defined in a physical saved segment that is a member of a CP segment space, *space* is the CP segment space.
- If *segname* is a skeleton segment (reserved with the SKELeton option), *space* is the same as *segname*.

location
is the starting address of the storage space.

length
is the length of the storage space.

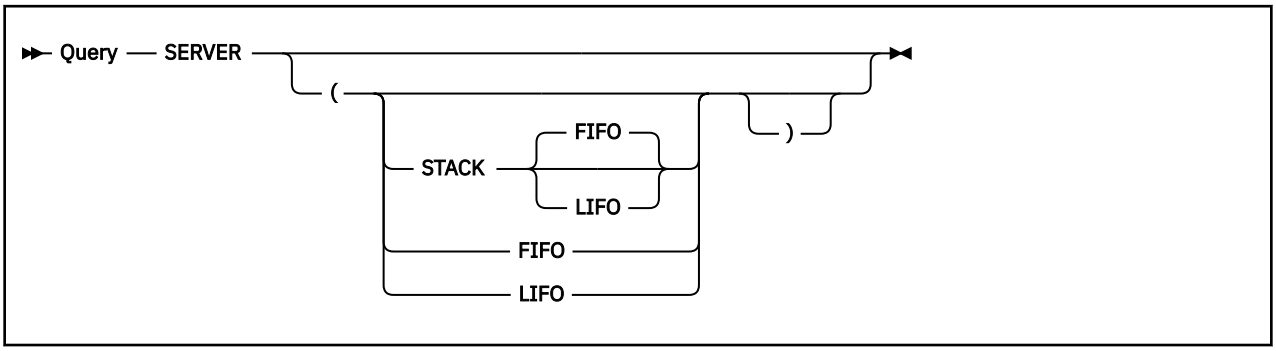
6. For

Returns information about all the storage spaces that contain or are reserved for saved segments. The response is the same as for QUERY SEGMENT *segname* SPACE; one line is returned for each segment storage space.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see “QUERY” on page 620.

QUERY SERVER



Authorization

General User

Purpose

Use the QUERY SERVER command to display whether the virtual machine will process or reject private resource conversation requests. Use the SET SERVER command to enable private resource processing. For more information, see "SET SERVER" on page 989.

Responses

```
SERVER      ON
```

or

```
SERVER      OFF
```

Where:

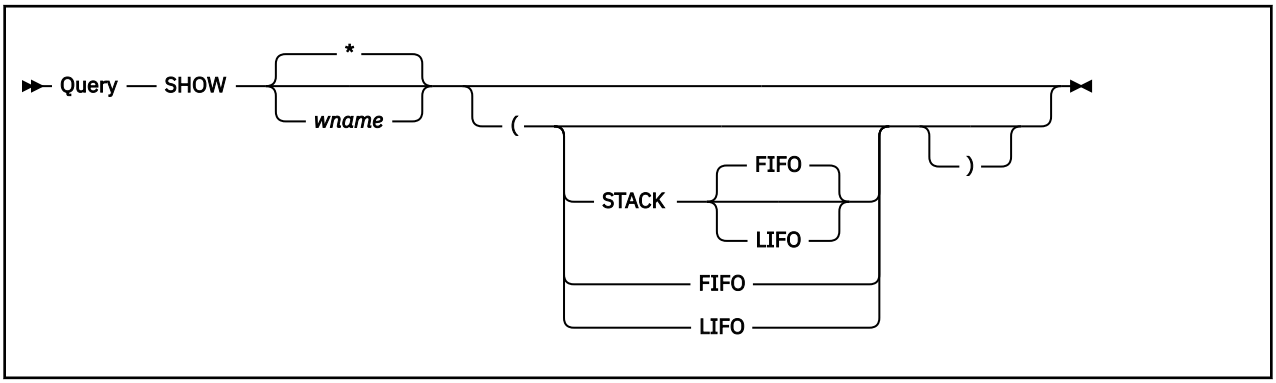
ON
indicates the virtual machine will process any private resource conversation requests.

OFF
indicates the virtual machine will reject any private resource conversation requests.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see "QUERY" on page 620.

QUERY SHOW



Authorization

General User

Purpose

Use the QUERY SHOW command to display information about one or all windows that are not hidden.

Operands

wname
is the name of the window about which information is to be displayed.

*****
specifies information for all windows is to be displayed in the order in which they are being displayed on the physical screen. This is the default.

Responses

One line is displayed for each visible window. If the specified window is not displayed, you will simply get an error message.

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.

```
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.

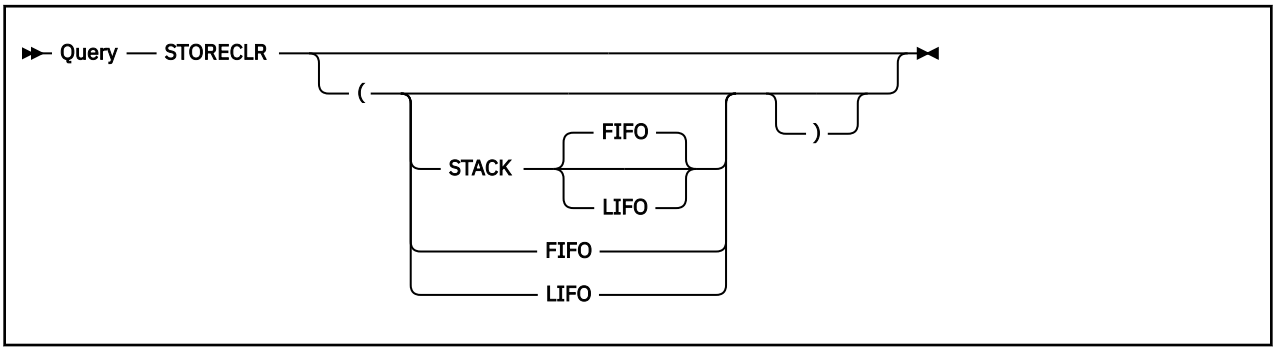
line
is the line number in the virtual screen where the window is connected.

column
is the column number in the virtual screen where the window is connected.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see “QUERY” on page 620.

QUERY STORECLR



Authorization

General User

Purpose

Use the QUERY STORECLR command to display the current setting of CMS GETMAIN free storage clean up. Use the SET STORECLR command to change the way CMS releases GETMAIN storage. For more information, see "SET STORECLR" on page 990.

Responses

```
STORECLR = ENDSVC
```

or

```
STORECLR = ENDCMD
```

Where:

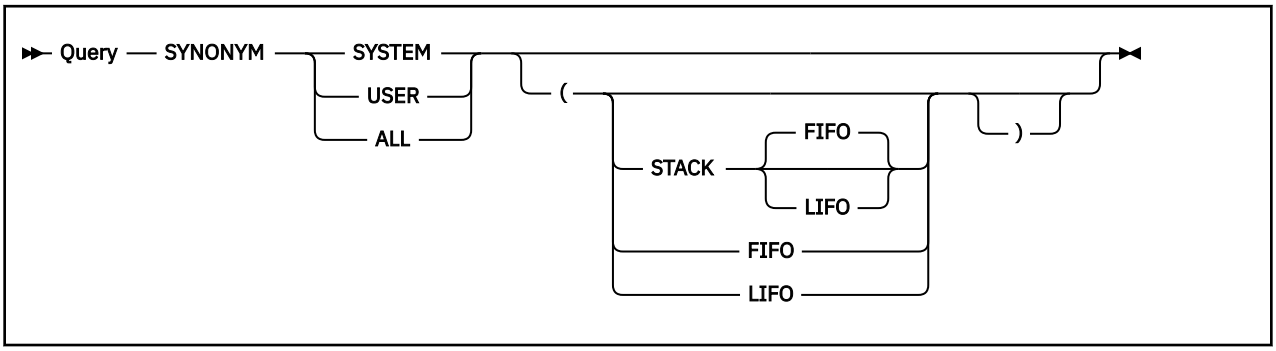
ENDSVC
indicates CMS releases GETMAIN storage at SVC 202, SVC 204/CMSCALL, and SVC 42/ATTACH termination. CMS treats the STRINIT macro as a NOP.

ENDCMD
indicates CMS returns GETMAIN storage at end-of-command, the point where the CMS ready message is displayed. CMS honors user invocations of STRINIT.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see "QUERY" on page 620.

QUERY SYNONYM



Authorization

General User

Purpose

Use the QUERY SYNONYM command to display the user and CMS system synonyms defined by the SYNONYM command.

Operands

SYSTEM
displays the CMS system synonyms in effect

USER
displays user synonyms in effect

ALL
displays all synonyms in effect

Responses

1. For


   ```
   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
   ```

2. For


   ```
   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
```

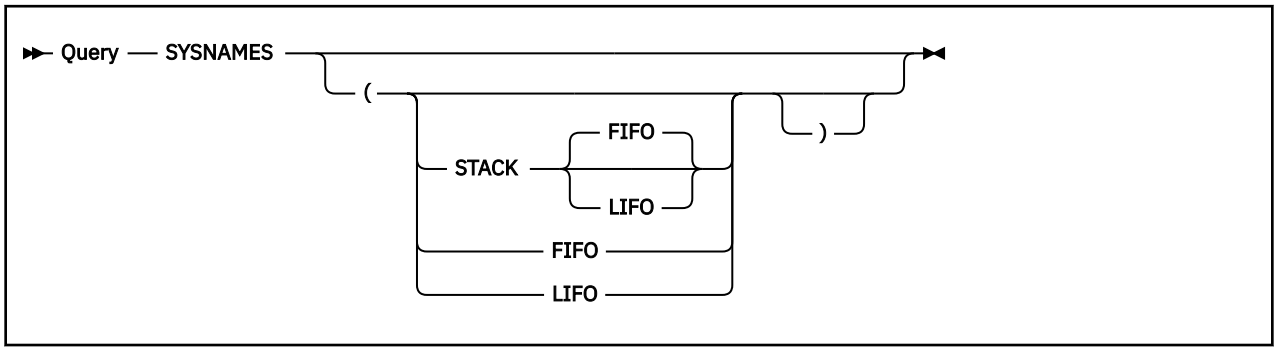
3. For

The response is that of QUERY SYNONYM SYSTEM followed by that of QUERY SYNONYM USER.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see “QUERY” on page 620.

QUERY SYSNAMES



Authorization

General User

Purpose

Use the QUERY SYSNAMES command to display the names of the standard saved systems. For more information on changing the SYSNAME table, see “SET SYSNAME” on page 992.

Responses

```
SYSNAMES:  CMSVSAM  CMSAMS   CMSDOS   CMSBAM
ENTRIES:   CMSVSAM  CMSAMS   CMSDOS   CMSBAM
```

Where:

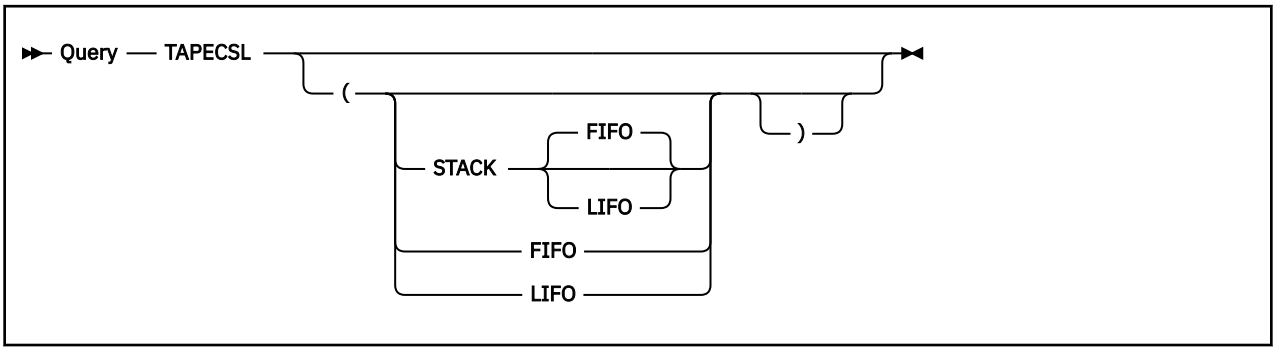
SYSNAMES
are the standard names that identify the discontiguous saved systems.

ENTRIES
are the standard system default names or the system names established through the SET SYSNAME command.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see “QUERY” on page 620.

QUERY TAPECSL



Authorization

General User

Purpose

Use the QUERY TAPECSL command to display the setting of CMS OS Simulation tape action for calling DFSMS/RMS via CSL for tape mounts or demounts on automated tape librarian controlled tape drives. For more information on how to change the setting, see "SET TAPECSL" on page 993.

Responses

```
TAPECSL = ON
```

or

```
TAPECSL = OFF
```

Where:

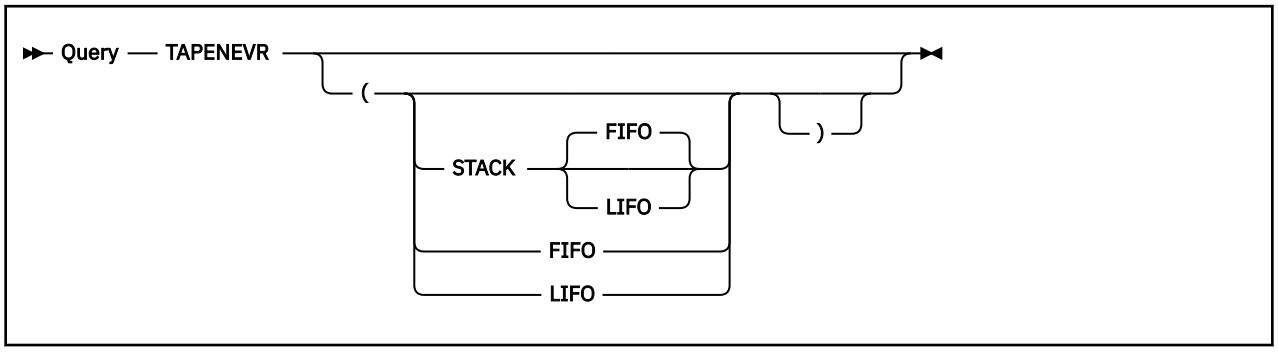
ON
the CMS OS Simulation routines will use CSL to call DFSMS/RMS functions to either mount or demount tapes on an automated tape librarian controlled tape drive. This is the default.

OFF
the CMS OS Simulation routines will not use CSL to call DFSMS/RMS functions to either mount or demount tapes on an automated tape librarian controlled tape drive. Instead, native CMS OS Simulation routines will issue messages to an operator to mount a tape and use the TAPE RUN command to unload a tape from the drive.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see "QUERY" on page 620.

QUERY TAPENEVR



Authorization

General User

Purpose

Use the QUERY TAPENEVR command to display the setting of CMS OS Simulation tape label date checking for 'Unexpired Files'. For information on how to change the setting, see “SET TAPENEVR” on page 994.

Responses

```
TAPENEVR =   ON

or

TAPENEVR =   OFF
```

Where:

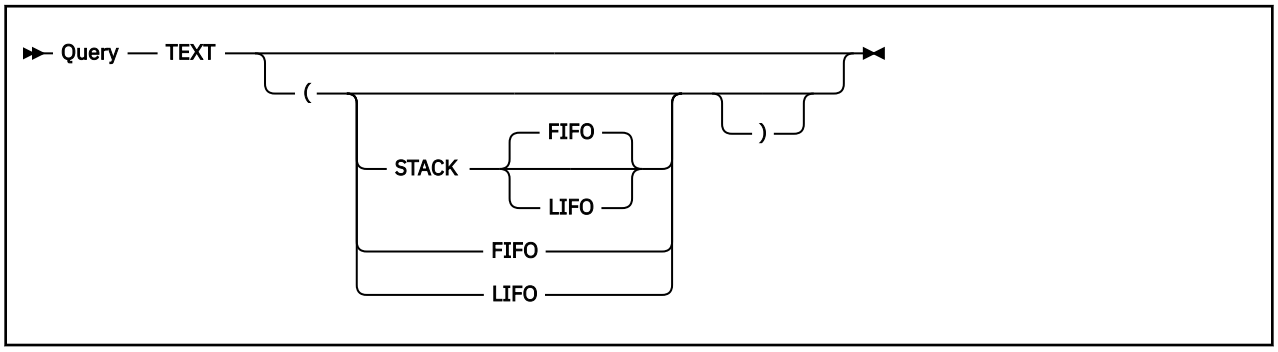
ON
indicates existing 'Never Expire' standard tape label Julian dates of ' 99365' and ' 99366' are honored as not expired by CMS OS Simulation tape label processing routines and an 'Unexpired File' prompt is issued to the user. This is the default.

OFF
indicates date values in standard tape labels are treated as normal dates. Existing 'Never Expire' date values are not honored as special values. The label date is tested against the current system date. If the label date is greater than the current date, CMS OS Simulation issues an 'Unexpired File' prompt. Otherwise, the label date is considered expired, and the tape is writable.

Options

For information on the Options, Usage Notes, and Messages that apply to all operands of the QUERY command, see “QUERY” on page 620.

QUERY TEXT



Authorization

General User

Purpose

Use the QUERY TEXT command to display the status of TEXT character code conversion. Use the SET TEXT command to activate and deactivate character code conversion. For more information, see "SET TEXT" on page 995.

Responses

```
TEXT      ON
```

or

```
TEXT      OFF
```

Where:

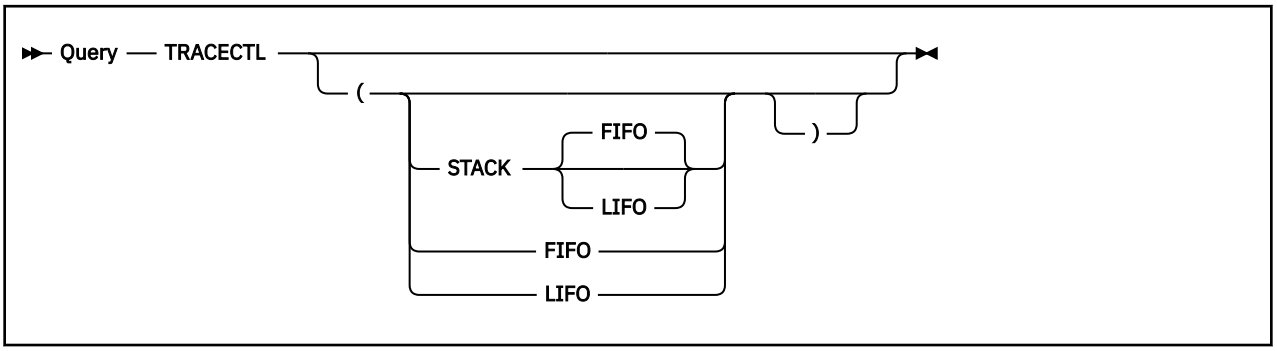
ON
indicates text character conversion is in effect.

OFF
indicates text character conversion is not in effect.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see "QUERY" on page 620.

QUERY TRACECTL



Authorization

General User

Purpose

Use the QUERY TRACECTL command to display a list of the events enabled for CMS internal tracing, as well as a wrapsize to show how many trace events are retained in storage. This command is useful for problem diagnosis in the CMS multitasking environment.

Responses

```
TRACECTL WRAPSIZE -1 COMM ON DISP ON PROC ON LANG ON SYNC ON MISC ON
```

or

```
TRACECTL WRAPSIZE 0 COMM OFF DISP OFF PROC OFF LANG OFF SYNC OFF MISC OFF
```

or a combination of the above:

```
TRACECTL WRAPSIZE 20 COMM ON DISP OFF PROC OFF LANG ON SYNC ON MISC OFF
```

Where:

COMM ON or OFF
indicates Communication event tracing is on or off.

DISP ON or OFF
indicates Dispatch event tracing is on or off.

PROC ON or OFF
indicates Process Management event tracing is on or off.

LANG ON or OFF
indicates Language-specific event tracing is on or off.

SYNC ON or OFF
indicates Synchronization event tracing is on or off.

MISC ON or OFF
indicates Miscellaneous event tracing is on or off.

WRAPSIZE *n*
indicates how many trace events are retained if no eligible trace event monitor exists at the time the event is signaled.

n
A positive integer value greater than 0. When the wrapsize is exceeded, the oldest trace event is discarded to make room for the newest arrival.

0
No trace events are being retained.

-1
Trace events continue to be retained until virtual storage is exhausted.

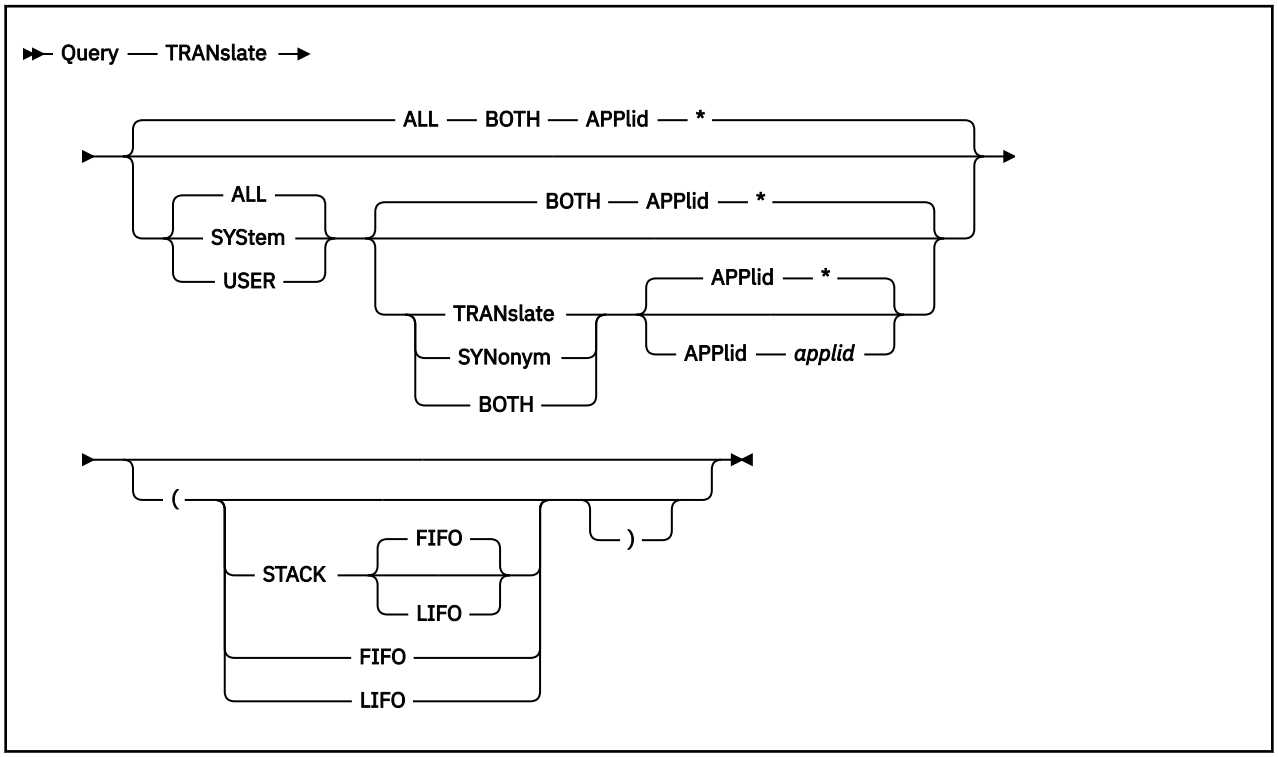
Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see "QUERY" on page 620.

Usage Notes

Use the the TRACECTL command to change the settings of the CMS internal trace events and the wrapsize. For more information, see "TRACECTL" on page 1082.

QUERY TRANSLATE



Authorization

General User

Purpose

Use the QUERY TRANSLATE command to display the translations and translation synonyms in effect. Use the SET TRANSLATE command to control the translations and synonyms. For more information, see "SET TRANSLATE" on page 998.

Operands

TRANslate
displays translations and translation synonyms 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.

Responses

The QUERY TRANSLATE command responds with both:

- One or two messages stating whether the translations and translation synonyms are active
- 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 this 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
     .                 .                    .
     .                 .                    .
     .                 .                    .
```

Examples

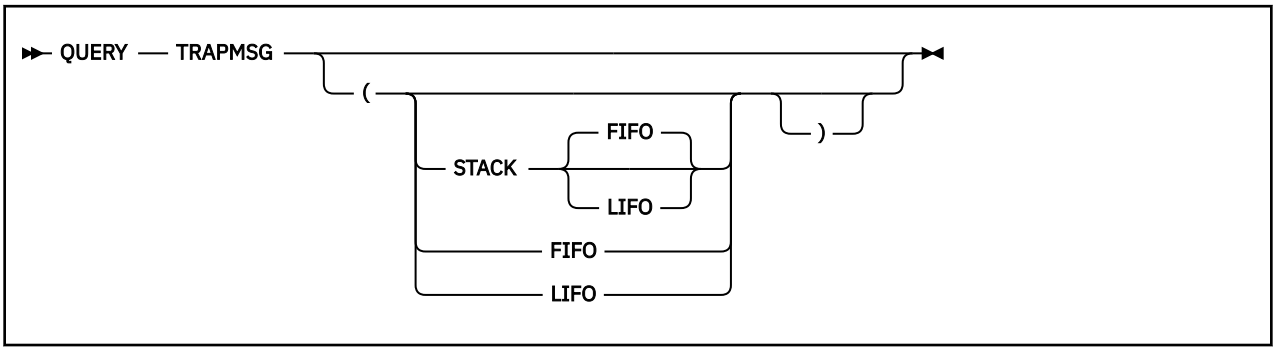
To display only the system translations in effect, enter:

```
query translate system translate
```

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see “QUERY” on page 620.

QUERY TRAPMSG



Authorization

General User

Purpose

Use the QUERY TRAPMSG command to look at a current TRAPMSG setting.

The message trap can be activated by issuing SET TRAPMSG ON with the respective message ID and dump range. It is used whenever a message is received that needs further explanation or debugging to find the cause of the message. SET TRAPMSG would generally be used only when contact of a support group for further analysis is necessary. For more information, see "SET TRAPMSG" on page 1000.

Responses

- If message trap is active, output given by the QUERY TRAPMSG command will indicate the message ID, dump range, FORMAT number, STOP, DCSS, DATASPACE and TO.

```
SET TRAPMSG ON 1234 0-END

QUERY TRAPMSG

APPLID CALLER MSG# LET hexloc1  hexloc2  TO       FMT     Other options
------ ------ ---- --- -------- -------- -------- --- -------------------------
 DMS    ???   1234  ?  00000000 017FFFFF *         *  NODCSS RUN NODATASPACE
```

```
SET TRAPMSG ON DMSTRP1234E 0-END

QUERY TRAPMSG

APPLID CALLER MSG# LET hexloc1  hexloc2  TO       FMT     Other options
------ ------ ---- --- -------- -------- -------- --- -------------------------
 DMS    TRP   1234  E  00000000 017FFFFF *         *  NODCSS RUN NODATASPACE
```

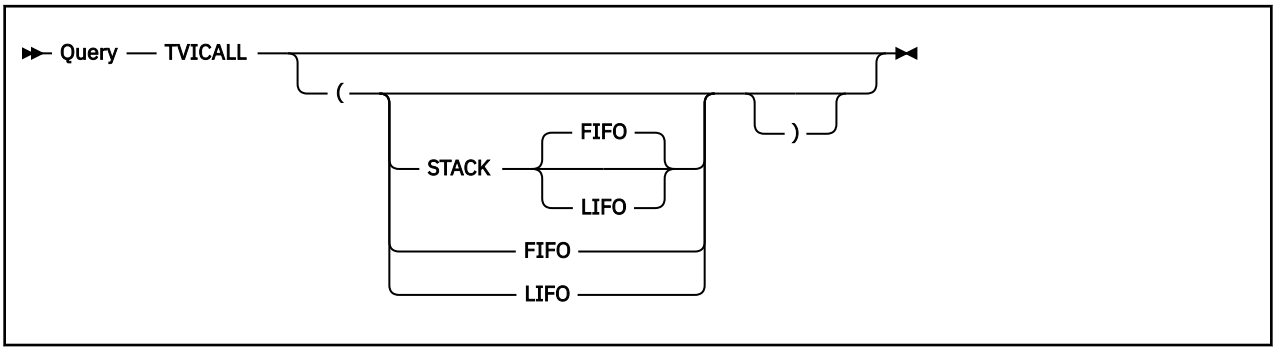
- If no message trap is active, it will return a message saying

```
TRAPMSG  = OFF
```

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see "QUERY" on page 620.

QUERY TVICALL



Authorization

General User

Purpose

Use the QUERY TVICALL command to return the current status setting of the SET TVICALL command. For more information, see "SET TVICALL" on page 1004.

Responses

```
TVICALL ALL
```

or

```
TVICALL STD
```

or

```
TVICALL OFF
```

Where:

ALL
the CMS OS Simulation routines will call the DMSTVI tape sub-system exit for any valid tape label type.

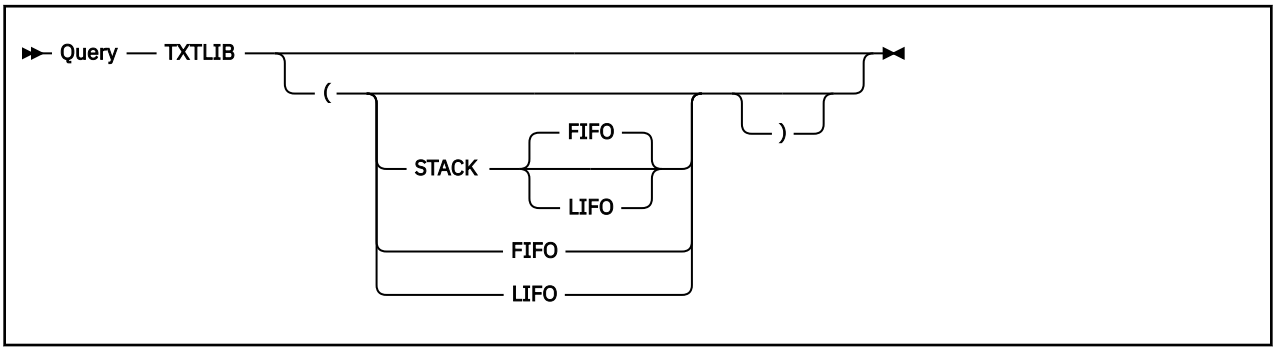
STD
the CMS OS Simulation routines will call the DMSTVI tape sub-system exit only for the tape standard label types (AL, AUL, SL, SUL).

OFF
the interface to the DMSTVI tape sub-system exit routine has been turned off. No calls for any tape processing will be made to it from the CMS OS Simulation routines.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see "QUERY" on page 620.

QUERY TXTLIB



Authorization

General User

Purpose

Use the QUERY TXTLIB command to display the names of all files, with a file type of TXTLIB, to be searched for unresolved references (that is, all TXTLIBs specified on the last GLOBAL TXTLIB command, if any).

Responses

```
TXTLIB = libname1 … libname8
  .          .            .
  .          .            .
  .          .            .
```

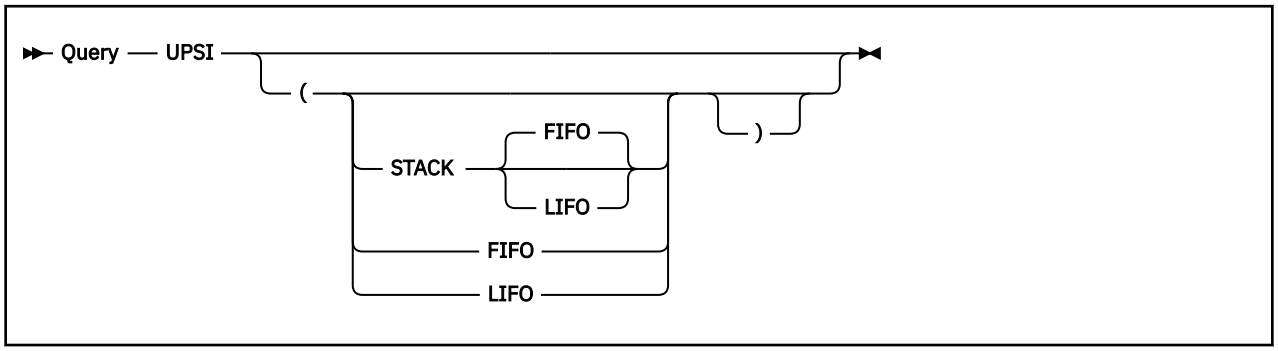
Up to eight names are displayed per line, for as many lines as necessary. If no TXTLIBs are to be searched for unresolved references, the following message is displayed at the terminal:

```
TXTLIB = NONE
```

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see “QUERY” on page 620.

QUERY UPSI



Authorization

General User

Purpose

Use the QUERY UPSI command to display the current setting of the UPSI byte in the CMS/DOS environment. The eight individual bits are displayed as zeros or ones, depending on whether the corresponding bit is on or off. Use the SET UPSI command to set the UPSI byte. For more information, see "SET UPSI" on page 1005.

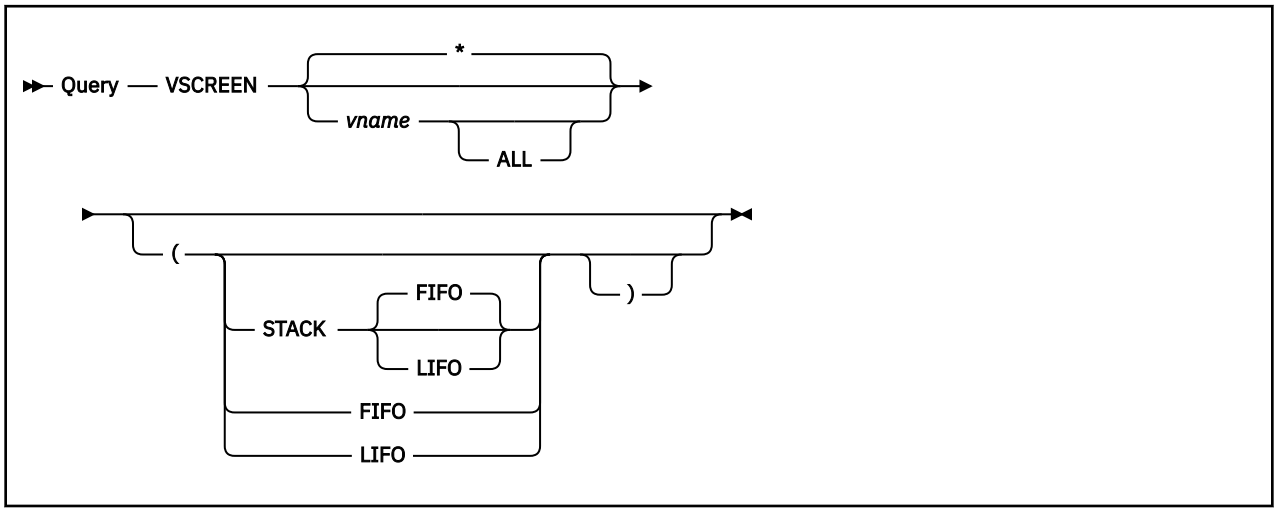
Responses

```
UPSI = nnnnnnnn
```

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see "QUERY" on page 620.

QUERY VSCREEN



Authorization

General User

Purpose

Use the QUERY VSCREEN command to display information about one or all of the virtual screens you have defined. Use the SET VSCREEN command to control how a virtual screen is updated with data. For more information, see "SET VSCREEN" on page 1006.

Operands

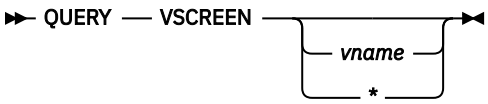
vname
is the name of the virtual screen about which information is to be displayed.

*****
indicates information about all virtual screens is to be displayed. This is the default.

ALL
specifies the attributes and extended attributes of one or all virtual screens is to be displayed in addition to the same information as QUERY VSCREEN.

Responses

1. For



One line is displayed for each virtual screen.

```
VSCREEN  vname   lines   cols   rtop   rbot
```

Where:

vname
is the name of the virtual screen.

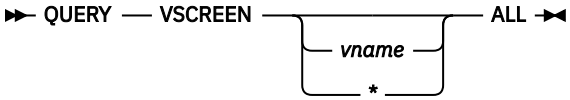
lines
is the number of lines in the virtual screen.

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.

2. For




```
VSCREEN vname lines cols rtop rbot (high protect color
exthi psset type system outline charset
```

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 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. It may be PS0, PS1, PS8, PSA, PSB, PSC, PSD, PSE, or PSF.

type
indicates data is moved to the virtual screen when the virtual screen queue is processed (TYPE) or 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.

outline
is the field outlining of the virtual screen. Outlining may be BOXDEF (the default outlining for the device), BOX (a full box), or any combination of the following to obtain the remaining 14 possible valid values (see the VSCREEN DEFINE command):

BOXLEFT
A vertical line on the left of the field

BOXOVER
Overline

BOXRIGHT
A vertical line on the right of the field

BOXUNDER
Underline

Note: The outlining values are returned in the order above. Also, if BOXLeft, BOXOver, BOXRight, and BOXUnder have all been specified when the vscreen was defined, QUERY displays the outlining value as BOX.

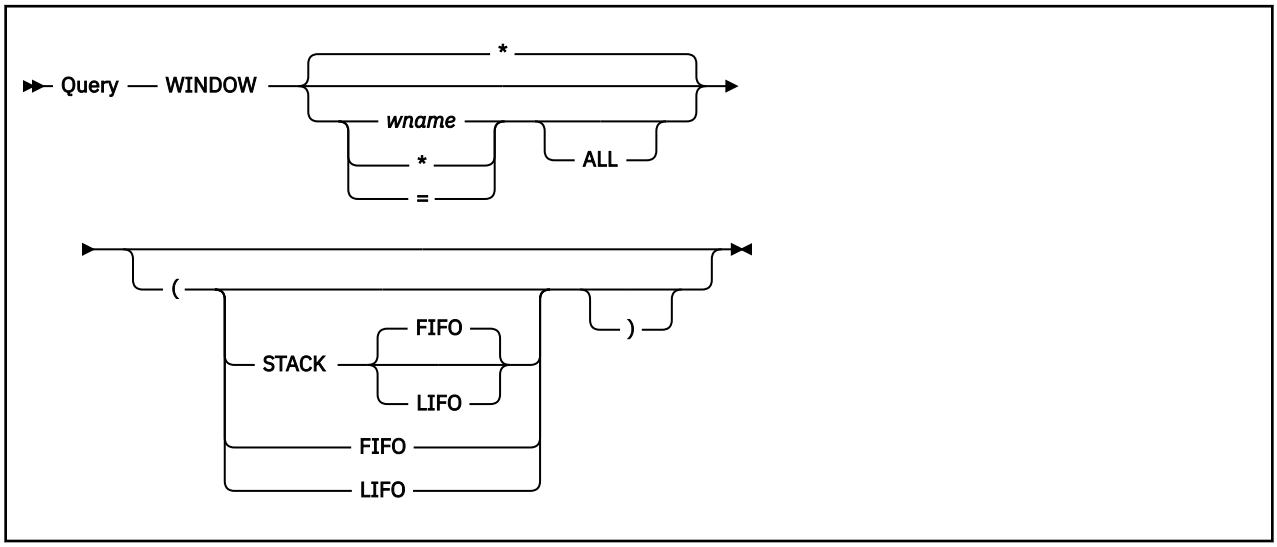
charset
indicates whether the virtual screen field is mixed DBCS or SBCS. It may be MIXED or SBCS.

Note: When using the ALL operand, the response may be too long for one line. If so, it wraps to the next one.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see "QUERY" on page 620.

QUERY WINDOW



Authorization

General User

Purpose

Use the QUERY WINDOW command to display information about one or all of the windows you have defined. Use the SET WINDOW command to specify window characteristics. For more information, see "SET WINDOW" on page 1009.

Operands

wname
is the name of the window about which information is to be displayed.

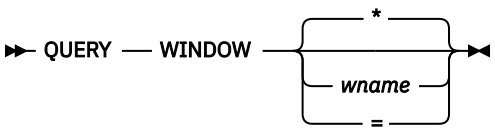
=
indicates information about the topmost window is to be displayed. Only a window defined with the TOP option can qualify as the topmost window.

*****
indicates information for all windows is to be displayed. This is the default.

ALL
displays the options for a window or all windows in addition to the size and location.

Responses

1. For



One line is displayed for each window.

```
WINDOW  wname   lines   cols   psline   pscol
```

Where:

wname
is the name of the window.

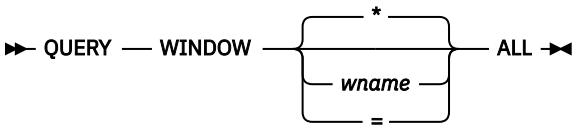
lines
is the number of lines in the window.

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.

2. For




```
WINDOW wname  lines  cols  psline  pscol (type
border  pop  top  system
```

Where:

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 there is no effect on the window's position in the ordered list of windows (NOPOP) when the virtual screen 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)

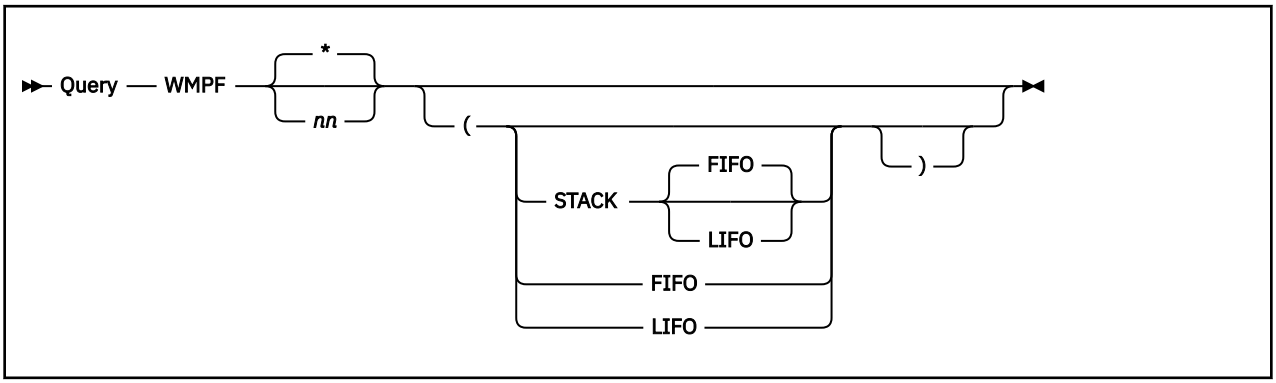
system
indicates whether the window is retained (SYSTEM) or deleted (USER) when a task abnormally ends (abend) or when the HX (halt execution) command is issued.

Note: When using the ALL operand, the response may be too long for one line. If so, it wraps to the next one.

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see “QUERY” on page 620.

QUERY WMPF



Authorization

General User

Purpose

Use the QUERY WMPF command to display the definition of one or all WMPF keys that were set with the SET WMPF command. Use the SET WMPF command to set the WMPF keys. For more information, see "SET WMPF" on page 1011.

Operands

nn
is the number (from 1 to 24) of the PF key whose definition is to be displayed.

***
indicates the definitions for all WMPF keys are to be displayed. This is the default.

Responses

For

One line is displayed for each WMPF key.

```
WMPF nn  pseudonym  keyword   string
```

Where:

pseudonym
is a 9-character representation 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.

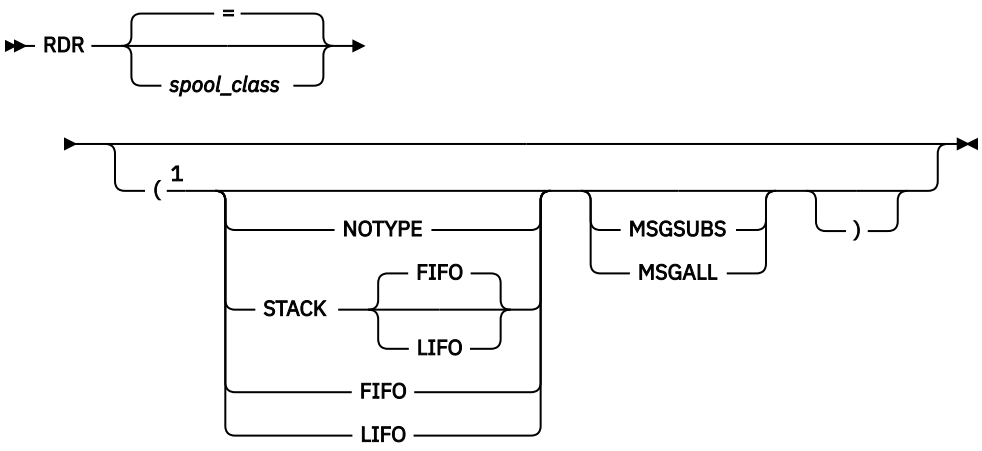
An undefined WMPF key is displayed as

```
WMPF nn
```

Options

For more information on the Options, Usage Notes, and Error Messages that apply to all operands of the QUERY command, see “QUERY” on page 620.

RDR



Notes:

[1] You can enter Options in any order between the parentheses.

Authorization

General User

Purpose

Use the RDR command 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 they are held.

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 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 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 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 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 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. For more information on substitution data, see "Usage Notes" on page 785.

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. For more information on substitution data, see "Usage Notes" on page 785.

Usage Notes

1. If the RDR command is issued and it finds a file printed to a VAFP printer, a return code of 38 will result with the following message:

   ```
   VAFP Printer file
   ```

2. Issued from the command line with no options, RDR displays the return code and message at the terminal.
3. If RDR is issued with the NOTYPE option, only the return code is returned to the terminal or calling program.
4. If no options are specified and RDR is issued from an EXEC, the message is displayed at the terminal and the return code is accessible from within the EXEC.
5. Issued with the NOTYPE option from a REXX exec, 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'
   ```

6. 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.
7. The RDR command changes the order of the files in your virtual reader. Files not held are re-ordered according to class.
8. The RDR command does not handle MONITOR files.
9. The substitution data line generated by the MSGSUBS and MSGALL options contains the message identifier of the actual message, followed by any available substitution data.

   The substitution data in a message is the variable information contained in it. For more information about substitution in a message, see *z/VM: CMS Application Development Guide*.

Responses

The messages you can receive from RDR, along with their associated return codes and meanings, are:

RC	Message	Explanation
0	Reader empty	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.
1	System dump file	The reader contains a system dump file, which can be handled using the appropriate system utility.

RC	Message	Explanation
2	PRINTER FILE (ITEM LENGTH *lrecl*)	The reader contains a printer file with a logical record length of *lrecl*.
3	DISK LOAD *fn ft fm*	The reader contains a file sent through DISK DUMP from an old CMS file system that supports minidisks formatted only in 800-byte physical blocks.
4	:READ *fn ft fm originid mm/dd/yy hh:mm:ss*	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, the result will be in the above format; otherwise, the READ control card will be displayed through column 72.
5	Cards for IPL	The reader contains a file that has IPL cards as the first cards in the file.
6	Unnamed card deck	The reader contains a PUNCH file that can not be identified.
7	:READ *fn ft fm*	The reader contains a file that is a printer file.
9	Reader not operational	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.)
10	File with X'5A' CCW	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.
13	Reader not ready	The reader is not ready. To reverse the not ready status, issue the CP command READY 00C.
18	Console spool file	The reader contains a file that is from a console.
22	DISK LOAD *fn ft fm*	The reader contains a file sent through DISK DUMP from a CMS file system or from the Shared File System (SFS).
23	Netdata file	The reader contains a file that was sent using the SENDFILE command with the NEW option.
26	Message	The reader contains a non-console or non-printer file that has a MSG control card as the first real record.
38	VAFP Printer file	The reader contains a VAFP printer file.

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

This example shows the response you will receive when you issue RDR (MSGSUBS against a printer file 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 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

This 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

These 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.

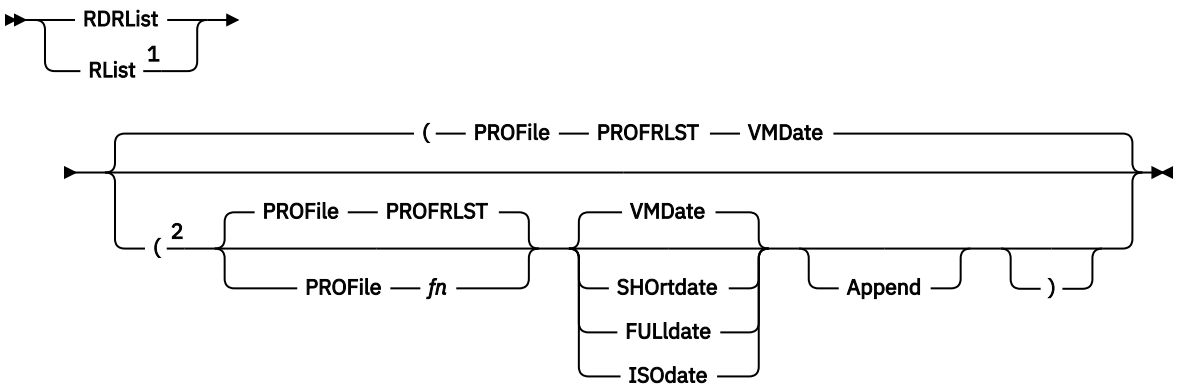
Messages and Return Codes

- DMS070E Invalid parameter *parameter* [RC=24]
- DMS630S Error accessing spool file [RC=36]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

RDRLIST



Notes:

[1] DISCARD and EXECUTE are special commands that can be used in the RDRLIST environment.
[2] You can enter Options in any order between the parentheses.

Authorization

General User

Purpose

Use the RDRLIST command to display information about the files in your virtual reader. The RDRLIST environment is controlled by XEDIT. 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 the files in your reader were sent to you by other computer users, on your computer or on other computers.

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 Usage Note “10” on page 793, "Default PF Key Settings".

VMDate
displays the creation date of each file in the format specified by the user's default date format setting. This is the default. For more information on the user's default date format setting, see Usage Note “16” on page 794.

SHOrtdate
displays the creation date of each file in *mm/dd* format.

Where:

mm
specifies the month

dd
specifies the day of the month

FULldate
displays the creation date of each file in *mm/dd/yyyy* format.

Where:

mm
specifies the month

dd
specifies the day of the month

yyyy
specifies the 4-digit year

ISOdate
displays the creation date of each file as *yyyy-mm-dd* format.

Where:

yyyy
specifies the 4-digit year

mm
specifies the month

dd
specifies the day of the month

Append
specifies the list of files in your reader should be appended to the existing list. This option has meaning only when entered from within RDRLIST. When entered outside of RDRLIST, it results in an error condition.

Usage Notes

1. For more information on how to customize this command see Appendix A, "Customizing Profiles for CMS Productivity Aids," on page 1419.
2. 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. For more information, see "EXECUTE" on page 1393. The DISCARD command allows you to purge the reader files displayed by RDRLIST. For more information, see "DISCARD" on page 1392.
3. Tailoring the RDRLIST Command Options

   You can use the DEFAULTS command to set up options and 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. For more information, see "DEFAULTS" on page 153.
4. 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 the file is held
   - Creation date and time
   - Originating user ID and node

A truncation algorithm is used for domain names on the RDRLIST panel. For example, if the first token is greater than 8 characters, the *userid* is truncated to the first 8 characters of the token. For more information on the truncation algorithm for domain names, see *z/VM: TCP/IP User's Guide*.

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, and so forth.

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.

5. 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.

6. 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 spool ID number is automatically appended to the end of the command.

   Note: When a 'B' is entered, the CP command search is bypassed. If passed to CP the 'B' would be interpreted as the CP BEGIN command. Because the spool ID is appended to the 'B' and the CP BEGIN command expects an address as input, the spool ID would be interpreted as such. In most cases this would cause an ABEND of the CMS virtual machine.

   Use the slash (/) symbols described below to specify the spool ID 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. When you are finished typing the command, erase the rest of the line by pressing Erase EOF, or space over the rest of the line. Press Enter. You may use the DELETE key to erase the rest of the line, but do not use it to erase only part of the rest of the line.

   For more information, see “EXECUTE” on page 1393.

7. A message from RDRLIST may sometimes appear on the line where a reader file is listed. To enter commands on that line, type over the response and press Enter.

   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
   identifies the user ID of the recipient.

   To transfer a reader file to a user on another computer connected to yours (remote), you must know the user ID of the user, the user ID (rscsid) of the virtual machine at your location 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
identifies the locid of the remote computer

USERB
identifies the user ID of the recipient

After you have entered the TAG command, on the same line type:

```
TRANSFER RDR / NET
```

Where:

NET
identifies the user ID (rscsid) of the virtual machine at your location running RSCS.

To purge a file, you would move the cursor to that line on the screen, and type "discard" in the space provided to the left of the file name. For more information, see “DISCARD” on page 1392. When you press Enter, 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. For more information, see “Responses” on page 796.

Commands typed on the RDRLIST command line are executed first, then PF key actions are taken, then commands typed in the prefix area are executed when you press `Enter`. 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.

8. 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. For more information, see “EXECUTE” on page 1393. For more information, see “Examples” on page 795. These symbols can be used:

   Symbol
   Description

   /
   means the spool ID 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.

9. Special Symbols Used Alone

These special symbols can be typed alone on the lines of the RDRLIST display:

Symbol
Description

=
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 Enter).

?
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.)

10. 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. Table 39 on page 793 shows the values to which the keys are set.

Table 39. Default Key Settings and Synonyms Set by PROFRLST XEDIT

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.)**
PF1	Help	Display RDRLIST command description.
PF2	Refresh	Update the list to indicate discarded files, and so forth.
PF3	Quit	Exit from RDRLIST display.
PF4	Sort	Sort by file type, file name.
PF5	Sort	Sort by date and time, in a calendar year, oldest to newest.
PF6	Sort	Sort by user ID, in alphabetic order.
PF7	Backward	Scroll back one screen.
PF8	Forward	Scroll forward one screen.
PF9	Receive	Receive the file pointed to by the cursor. For more information, see “RECEIVE” on page 806.
PF10		Not assigned.
PF11	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. For more information, see “PEEK” on page 595.
PF12	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 these synonyms on the RDRLIST command line:

Synonym
Description

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 by month, day, and time (oldest to most recent).

11. 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 PF11, which is set to the PEEK command. For more information on the PEEK screen, see "PEEK" on page 595.

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

    ```
    exec rdrlist
    ```

13. When you press PF9 (RECEIVE), you may receive prompting messages that require a response. For more information, see "RECEIVE" on page 806.

14. To obtain secure origin data for a spool file sent from a remote site using secure origin data support, the CMS RDRLIST EXEC issues DIAGNOSE X'F8' subcode X'01' instead of the CP TAG QUERY FILE command. However, if secure origin data support is not installed in the system or if a file arrives from another system that does not have this support, RDRLIST obtains the origin data as it does now—using the CP TAG QUERY FILE command.

    The RDRLIST command calls the EXECUTE XEDIT macro to refresh the RDRLIST panel every time a command is executed from it. That's why EXECUTE issues DIAGNOSE X'F8' subcode X'01' to obtain the real origin data of the spool file.

15. Open files are displayed on the RDRLIST screen with "OPEN" in the date field and "000C" in the time field. Keep in mind the origin user ID and node for an open spool file from a remote node will display the Networking Virtual Machine in the user ID field and an asterisk (*) in the node field.

16. If a date format is not specified on the RDRLIST command, the default date format is determined by the setting of the DEFAULTS command. The DEFAULTS command default (the initial setting) is VMDATE, which indicates the user's default date format is to be used. The DEFAULTS command setting can be changed by using DEFAULTS SET.

    The default date format for certain CP and CMS commands can be set on a system-wide basis and also for the individual user. The system-wide default date format is set with the SYSTEM_DATEFORMAT system configuration statement. The user's default date format is set with the DATEFORMAT user directory control statement. The system-wide default and the user's default can also be set with the CP SET DATEFORMAT command. If SET DATEFORMAT is not specified for an individual user and there is no DATEFORMAT statement in the user's directory entry, the default format for that user is the system-wide default.The system-wide and user settings can be queried with the CP QUERY DATEFORMAT command.

The hierarchy of possible date format settings for the RDRLIST command, from highest priority to lowest, is:

- RDRLIST command option
- DEFAULTS command setting or default
- User default
- System-wide default

17. When FULLDATE or ISODATE is in effect, the columns to the right of the Date field will be shifted to the right to accommodate the addition of the 4-digit year.
18. An existing RDRLIST profile (PROFRLST XEDIT) on your A-disk or in your search order may cause the files displayed by RDRLIST to be sorted incorrectly when sorted by date with FULLDATE or ISODATE in effect. To ensure the date sort function operates correctly, you should erase your old profile and build a new user profile that first calls the system profile for RDRLIST (PROFFLST) followed by your customized changes.
19. In an SSI cluster, when a single-configuration virtual machine issues the RDRLIST command, all spool files that reside in the shared spool of the cluster (except files transmitted through RSCS) will appear to be local. That is, the files will appear to originate from the member where the RDRLIST command was issued, although the files might have been created on other members. (For files transmitted through RSCS, the actual origin node will be displayed.)

    If the RDRLIST command is issued by a logon instance of a multiconfiguration virtual machine, only spool files created on that member (or transmitted through RSCS) will be displayed.

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 when an external security manager is not installed.

```
 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 43. Sample RDRLIST Screen

The appearance of the RDRLIST screen changes when an external security manager is called on to check security labels.

In the following sample screen, notice some fields are masked by asterisks. This indicates the security label governing your current session does not dominate that reader file. Consequently, the system does not allow you to observe these fields.

To see these fields, log off and log on again with a security label that dominates the reader file you want to work with.

```
 JSMITH   RDRLIST     A0  V 108  Trunc=108 Size=5  Line=1 Col=1 Alt=1
Cmd     Filename Filetype Class User At Node   Hold  Records  Date  Time
        PROJA    PLANS    PUN A PJONES  NODE7  NONE     9410  9/22  3:30:57
        PROJD    PLANS    PUN A MBROWN  NODE9  NONE    11074  9/26  5:35:21
        PROJE    NOTE     PUN A FGREEN  NODE8  NONE       67  9/22  3:30:27
        ******** ******** *** A KWHITE  NODE4  NONE  *******  9/26  5:35:10
        PROJB    MEMO     PUN A PJONES  NODE4  NONE       32  9/12  8:30:21









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 44. Sample RDRLIST Screen with an External Security Manager Installed

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 ENTER 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 spool ID 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 *z/VM: 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.
* spoolfn spoolft ** Command not issued, RDRLIST line has been changed **
```

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.
```

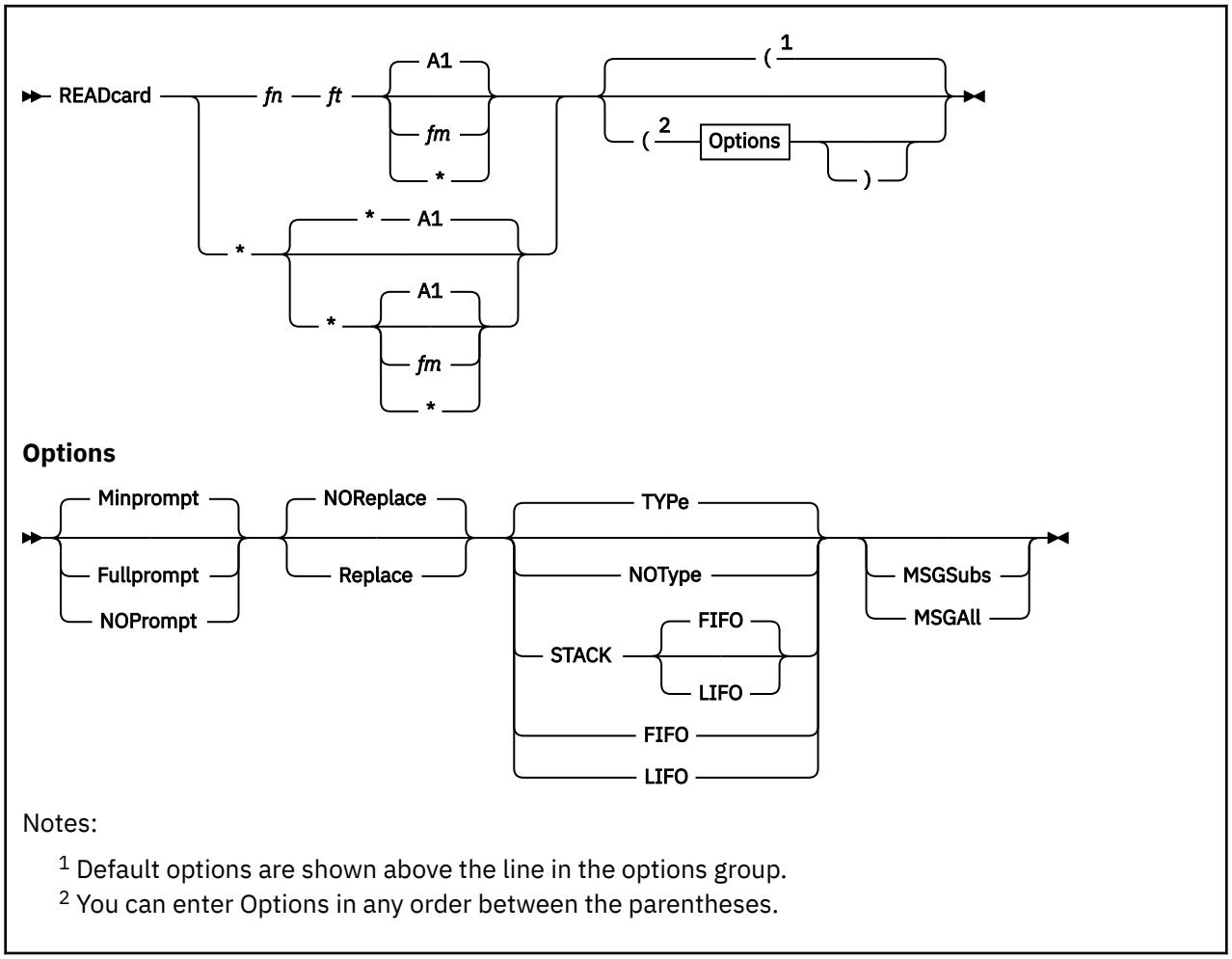
Messages and Return Codes

- DMS205E No files in your reader [RC=28]
- DMS651E APPEND must be issued from RDRLIST [RC=40]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

READCARD



Notes:

[1] Default options are shown above the line in the options group.
[2] You can enter Options in any order between the parentheses.

Authorization

General User

Purpose

Use the READCARD command to read data records from your virtual reader and to create CMS files containing the data records.

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.

***
indicates file identifiers are to be assigned according to READ control cards in the input deck. If an asterisk is specified for the file name, the file type defaults to an asterisk.

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 overrides the file mode letter and number on the control card and assigns that file mode to the file.

Options

Fullprompt
specifies a prompt is issued for each file.

Minprompt
specifies 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. The default is MINPROMPT.

NOPrompt
specifies 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 a file is not received that would overlay an existing file on the receiving disk or directory. The default is NOREPLACE.

TYPe
specifies responses (listed in response "9" on page 804 in "Responses" on page 803) will be displayed to the terminal.

NOType
specifies responses (excepting prompting messages) will not be displayed to the terminal.

FIFO
STACK
STACK FIFO
specifies responses (listed in response "9" on page 804 in "Responses" on page 803) will be placed in the stack in the order in which they would have been displayed.

Note: FIFO, STACK and STACK FIFO are synonymous.

LIFO
STACK LIFO
specifies responses (listed in response "9" on page 804 in "Responses" on page 803) will be placed in the stack in the inverse order in which they would have been displayed.

Note: LIFO and STACK LIFO are synonymous.

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. For more information on substitution data, see "Usage Notes" on page 800.

The lines are displayed or stacked in accordance with the FIFO, LIFO, NOTYPE, STACK, or TYPE 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 FIFO, LIFO, NOTYPE, STACK, or TYPE options.

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. If they are not, they will be made fixed-length and truncated to 80 characters.
2. Tailoring the READCARD Command Options

   You can use the DEFAULTS command to set up options and 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. For more information, see "DEFAULTS" on page 153.
3. If you specify the FULLPROMPT or MINPROMPT option, the valid responses include one of the following:

   - Digits specified in the prompt
   - Parenthetical words that follow a digit or any initial truncation of the word

   The meanings of these responses are:

   Response
   Description

   0 or No
   The file is not received and prompting continues for the next file. After the last file, the command is ended.

   1 or Yes
   Receives the file under the name *fn1 ft1 fm1* (or *fn3 ft3 fm3*).

   2 or Quit
   Ends the command.

   3 or Rename
   Requests prompt message DMS1080R so the incoming file can be received using a different name.
4. If you receive prompt message DMS1081R the valid responses include one of the following:

   - Digits specified in the prompt
   - Parenthetical words that follow 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 *fn ft fm* and repeats the original prompt message DMS1080R. 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.
5. 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 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.

   If the first card in the deck is not a READ control card, 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.

6. 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. The file mode will be A1, unless a different file mode was specified on the READCARD command.

   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).
7. 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.
8. 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:

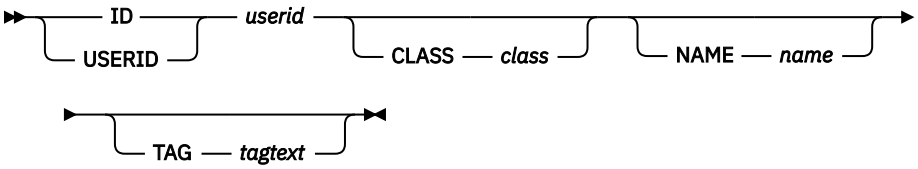
   ```
   readcard * * c
   ```
9. If you are preparing real or virtual card decks to send to your own or another user's virtual card reader, you may insert READ control cards to designate file names, file types, and file modes to be assigned to the files.

   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, because the mode letter is determined by specifications on the READCARD command line.
10. 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 and read the cards into the system card reader.

    The format of a CP ID card is:

    The ID or USERID keyword must begin in column 1, and each keyword and operand must be separated by at least one blank. *userid* is the user to receive the spool file containing the card deck, *class* and *name* are optional values that can be assigned to the spool file, and *tagtext* is optional data that will be associated with the spool file. If *tagtext* is specified, it must be the last operand on the card.
11. If the reader file being processed contains carriage control characters, the READCARD command returns the records with the carriage control characters stripped off.
12. 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.

13. You can read multiple spool files as one file by spooling your reader continuous (CP SPOOL command with the CONT option) and issuing the READCARD *file name file type* form of the command. The file on disk or directory will consist of multiple spool files separated by any READ control cards that were in the spool files.

    You cannot read multiple spool files as one file by spooling your reader continuous and issuing the READCARD * form of the command, as only one spool file will be read.

14. READCARD reads empty files based on the file attribute data sent with PUNCH. If you try to read an empty file into a minidisk, you will get this message:

    ```
    File fileid is empty; minidisk does not
    support empty files
    ```

    The file is not purged from your reader. You can DISCARD or PURGE the file from your reader or you can try to receive it into an SFS directory. If you receive the file into an SFS directory, the following will be transferred to the new file:

    - Record length
    - Record format
    - File ID

15. If you specify the MSGSUBS or MSGALL option, the READCARD command uses the following response format when returning the substitution data. This response has a fixed format and fixed length. The length is 69 characters.

    Note: Each individual field in the substitution line is initialized to an '*' followed by enough blanks to fill the field.

    The format is (each field is delimited by one blank):

    ```
    msgid fileid sentname oldfileid
    ```

    Where:

    msgid
    : the six character message identifier (four character message number and two character message format).

    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.

      This is the name of the file that was received.

    sentname
    : 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.

      This is the name of the file as specified by the sender. This field will only be supplied if it differs from *fileid*.

    oldfileid
    : 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.

      This is the name of the file that was replaced by *fileid*. This field will only be supplied if it differs from *fileid*. The only difference between the values of *fileid* and *oldfileid* will be the values of the *fm* fields.

    Note: The MSGALL and MSGSUBS options only apply to those responses controlled by the FIFO, LIFO, NOTYPE, STACK, and TYPE options.

16. If you specify a file name and file type and either the MSGALL or MSGSUBS option on the READCARD command, for example:

```
readcard junk file (MSGALL
```

The *sentname* and *oldfileid* fields of the substitution data response will always be '*'.

17. If READCARD is issued against a file printed to a VAFP printer, only the first 204 characters of each line will be read.

Responses

READCARD issues the following responses, depending on the situation.

1. If READCARD * was issued and prompting is *not* in effect, this response indicates a record beginning with :READ has been found in the spool file and the file ID is not valid:

```
DMS702E Missing, invalid, or incomplete fileid
in following READ control card:
:READ...
Command terminated
```

2. If READCARD * was issued and a control card was encountered in the input card stream, this response indicates the names assigned to each file:

```
DMS702I  :READ...
```

3. 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:

```
DMS702I  READ control card missing. Following
assumed:
:READ READCARD CMSUT1 A1
```

4. If READCARD * was issued and prompting is in effect, this response indicates a record beginning with :READ has been found in the spool file and the file ID is not valid:

```
DMS702W  Missing, invalid, or incomplete fileid
in following READ control card:
:READ...
Fileid changed to READCARD CMSUT1
```

5. If the records being read are not 80 bytes long, this message gives the length:

```
DMS738I  Record length is nnn bytes
```

6. If you specify the FULLPROMPT or MINPROMPT option one of these prompts will be displayed:

```
DMS1079R  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 you may specify when the name differs from the name of the incoming file.

7. If you respond with a 3 (or RENAME) to prompt message DMS1079R, the following message appears and you must enter a file ID in the form *fn* [*ft* [*fm*]].

   ```
   DMS1080R  Enter the new name for fn ft fm
   ```

8. If you respond to prompt message DMS1080R with a file ID that names an existing file, you receive this prompt:

   ```
   DMS1081R  Replace fn ft fm?
   Reply 0 (NO), 1 (YES), or 2 (QUIT)
   ```

9. 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
     ```

10. 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
    ```

    For more information about the use of this symbol, see “RDRLIST” on page 789.

Messages and Return Codes

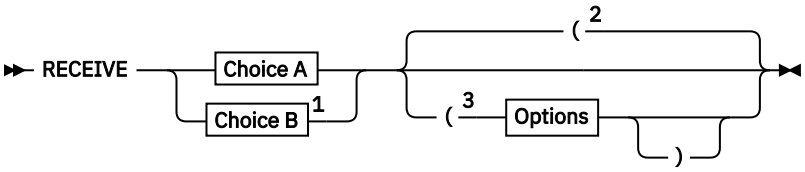
- DMS008E Device *vdev* {invalid or nonexistent|is an unsupported device type} [RC=36]

- DMS024E File *fn*[*ft fm*] already exists[; specify REPLACE option] [RC=28]
- DMS037E Filemode *mode*[(*vdev*)] is accessed as read/only [RC=36]
- DMS042E No fileid specified [RC=24]
- DMS054E Incomplete fileid specified [RC=24]
- DMS062E Invalid * in fileid [RC=20]
- DMS069E Filemode *mode* not accessed [RC=36]
- DMS105S Error *nn* writing file *fn ft fm* on disk or directory [RC=100]
- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS124S Error reading card file [RC=100]
- DMS205W Reader empty, reader not ready or empty reader file [RC=8|74]
- DMS257T Internal system error at address *address* (offset *offset*)
- DMS639E Error in *routine* routine; return code was *nnnn* [RC=*nnnn*]
- DMS671E Error loading [file] *fn ft fm*; rc=*nn* from RENAME [RC=100]
- DMS701W File *fileid* is empty; minidisk|filepool *filepoolid* does not support empty files [RC=74|88]
- DMS1123E Unknown response *text* ignored
- DMS1124W Spool file *spoolid* has been left in your reader because one or more files were not received [RC=1]
- DMS1138E Filesharing conflict [involving file *fn ft fm*] [RC=70]
- DMS1262S Error *nnn* opening file *fn ft fm* [RC=31 | 55 | 70 | 99 | 100]
- DMS1262S Error *nnn* closing file *fn ft fm* [RC=31 | 100]
- DMS1285S Default option *text* is invalid [RC=24]

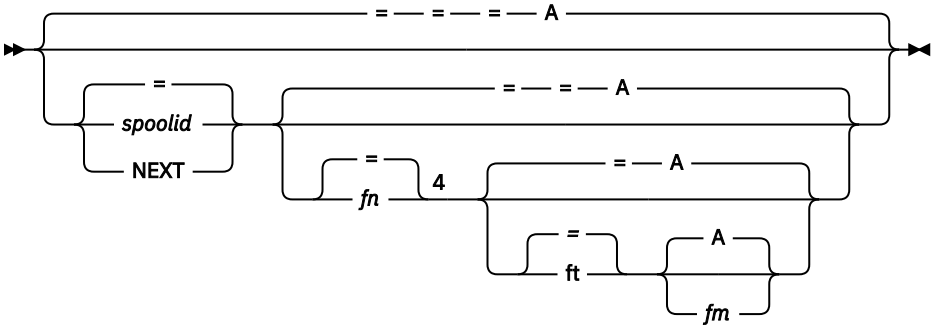
Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

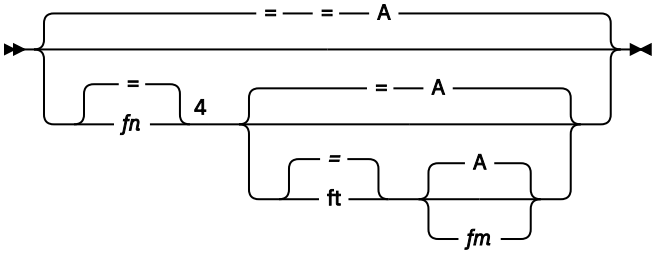
RECEIVE



Choice A



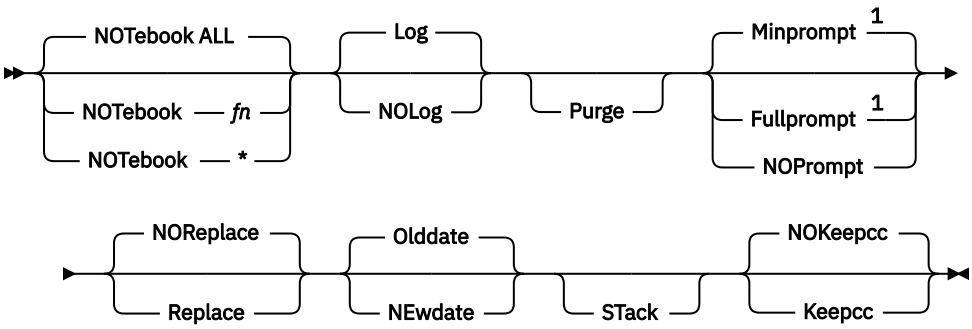
Choice B



Notes:

[1] This format of the command is valid only in the PEEK or RDRLIST environments.
[2] The defaults you receive appear above the line in the Options fragment.
[3] Options entered between the left '(' and right ')' delimiters can be in any order.
[4] *fn*, *ft*, and *fm* are ignored if the file was a note prepared with the NOTE command, PROFS, or the OfficeVision note facility.

Options



Notes:

[1] FULLPROMPT and MINPROMPT are ignored if the file being received is a NOTE.

Authorization

General User

Purpose

Use the RECEIVE command to read onto a disk or directory one of the files or notes in your virtual reader. In most cases these files were sent to you by other computer users, on your computer or on other computers.

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 they are held.

To give the file a new file identifier, you must specify the spool ID, '=', 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 A is accessed read-only, the default is the first read/write disk or directory in the search order.

If the file being received is a note (OfficeVision note facility), 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 spool ID and the PURGE option) are ignored. For more information, see Usage Note "9" on page 809, "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. The file mode of ALL NOTEBOOK is that of the first disk accessed read/write in your search order. 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, PROFS, or the OfficeVision note facility), this option is ignored.

For more information on the relationship between a "*userid* NAMES" file and the NOTEBOOK file, see "NAMEFIND" on page 516 and "NAMES" on page 535.

NOTebook *
specifies note is to be saved in a file named "*name* NOTEBOOK". To determine the value of *name*, your "*userid* NAMES" file is searched first for an entry that matches the sender's user ID and node ID. If a matching entry is:

- Found and it contains a notebook file name, that file name is used.

- Found but it does not contain a notebook file name, the sender's nickname is used.
- **Not** found, the sender's user ID is used.

If the file is not a note prepared using the NOTE command, this option is ignored.

Log
specifies 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. This is the default.

NOLog
specifies this file transmission is not to be logged.

Purge
specifies this file is to be purged and not read onto a disk or directory.

Fullprompt
specifies 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 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. The default is MINPROMPT.

If the file being received is a NOTE file, MINPROMPT is ignored.

NOPrompt
specifies 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 a file is not to be received that would overlay an existing file on the receiving disk or directory. The default is NOREPLACE.

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 UTC (Coordinated Universal Time). For more information, see Usage Note “16” on page 811, "Receiving NETDATA Files Using the Olddate Option". This is the default.

NEwdate
means to re-date the file to the current date and time it is received.

STack
specifies the message returned when RECEIVE completes successfully should be stacked (LIFO). If this option is not specified, the messages from RECEIVE (DMSWRC) are displayed at the terminal. (Any error messages that may be issued from NETDATA (DMSDDL), which is invoked by RECEIVE, will be stacked.)

NOKeepcc
Do not keep carriage control characters when receiving PRINT files. This is the default.

Keepcc
Keep carriage control characters when receiving PRINT files. When using the KEEPCC option, these files are stored with the carriage control characters -, +, 0-9, A, B, C, and blank.

Usage Notes

1. The RECEIVE command always receives the first file in the reader when entered from the command line in RDRLIST. This is because this form of the RECEIVE command uses the first parameter as the file name of the file to create, not as the spoolid of the file to receive.
2. The screen must be refreshed after RECEIVE is entered from the RDRLIST command line before RECEIVE can be issued again from the RDRLIST command line. If the screen is not refreshed, an error identifying the spoolid of the first file will be displayed.
3. If RECEIVE is issued with the REPLACE option, you will not be prompted about any file that may be overwritten.
4. Tailoring the RECEIVE Command Options

   You can use the DEFAULTS command to set up options and 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. For more information, see "DEFAULTS" on page 153.
5. Why Should I Use RECEIVE Instead of READCARD or DISK?

   You should use RECEIVE instead of READCARD or DISK for general purpose use because RECEIVE calls the appropriate CMS command to handle the spool file format. It also handles notes, acknowledgments, and so forth. In fact, RECEIVE handles most of the various formats of files that can appear in your virtual reader.

   Note: When using the NOKEEPCC option, RECEIVE strips records of carriage control characters. You can override this by using the KEEPCC option.
6. If RECEIVE is issued against a file printed to a VAFP printer, you will only receive the first 204 characters of each line in the file.

   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 sends 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. Also, if you have a DISK DUMP spool file with multiple files on it, issuing RECEIVE with a file mode other than A specified will cause all the files to be received on the specified file mode.
7. If the RECEIVE command is issued against a file printed to a VAFP printer with the PURGE option, the file will be purged.
8. 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.
9. Acknowledgments

   Acknowledgments can be sent to users on different computers connected by the RSCS network so they can be sure a file they sent was received.

   The sender can specify on the SENDFILE or NOTE command an acknowledgment be returned to them when a file is received. The SENDFILE command must be issued with the NEW option (the default) 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 "RDRLIST" on page 789. 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 spool ID and the PURGE option) are ignored. The acknowledgment is used to make an entry in your *userid* NETLOG file. This entry confirms the file you sent was received (or discarded). For more information on the format of entries in the *userid* NETLOG file, see "Examples" on page 812.

10. Responding to Prompting Messages

    If you specify the FULLPROMPT or MINPROMPT option the valid responses include one of the following:

    - Digits specified in the prompt
    - Parenthetical words that follow 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 *fn1 ft1 fm1* (or *fn3 ft3 fm3*).

    2 or Quit
    Ends the command.

    3 or Rename
    Requests prompt message DMS1080R so the incoming file can be received using a different name.

11. If you receive prompt message DMS1081R, the valid responses include one of the following:

    - Digits specified in the prompt
    - Parenthetical words that follow 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 *fn ft fm* and repeats the original prompt message DMS1080R. 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.

12. Which Prompt is Most Useful for You? If you:

    - Do issue either a QUERY RDR command or a RDRLIST command specify FULLPROMPT.
    - Issue a QUERY RDR command before issuing the RECEIVE command or if you issue RECEIVE from a RDRLIST screen specify MINPROMPT.
    - Issue the RECEIVE command in a controlled environment where the identity of all incoming files are known, specify NOPROMPT.

13. Special NETDATA Files from MVS with TSO Extensions Licensed Program

    The MVS with TSO Extensions licensed program can send an empty file, in which case RECEIVE will give you an error message indicating 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.

14. How does RECEIVE Determine the File ID?

    If you do not specify an *fn ft fm*, RECEIVE determines the file ID according to the method used to send the file.

File Format	How the FILE ID is Determined
NETDATA files	Uses the file ID specified on the SENDFILE command.
DISK DUMP files	Uses the file ID specified on the DISK DUMP command.
CONSOLE, PUNCH, and PRINTER files	Uses the fn and ft from the QUERY RDR ALL response.

    If a CP CHANGE command is used to change the file ID of a spooled file, only CONSOLE, PUNCH, and PRINTER files will be received under the name specified on the CP CHANGE command, regardless of the file ID specified on the imbedded ":READ" card, if any.

    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 in your virtual reader, use the RDR command.

15. The RECEIVE command does not handle MONITOR files or files with a SPECIAL status of YES. (The SPECIAL status indicates whether the file contains records with X'5A' carriage control characters. For more information on how to determine the SPECIAL status of a file, see the CP QUERY command in *z/VM: CP Commands and Utilities Reference*.

16. Receiving NETDATA Files Using the Olddate 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 UTC (Coordinated Universal Time).

    For example, suppose a file was last modified on 01/01/86 14:00 in a time zone 8 hours west of UTC, and it is sent using SENDFILE to a time zone 5 hours west of UTC. When the file is received, the time and date is changed to 01/01/86 17:00.

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

```
exec receive
```

18. 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.

19. If you choose to receive onto an SFS directory an empty file that has been disk dumped that directory must support empty files, and you must have it accessed.

20. PRINT files for KEEPCC option

    The PRINT files referred to in the KEEPCC option are printer files (return code 2 from RDR) and VAFP printer files (return code 38 from RDR). When receiving one of these files with the KEEPCC option, the record format will be V.

21. In an SSI cluster, any spool file received from the shared spool of the cluster (except a file transmitted through RSCS) will be recorded in the NETLOG file as originating from the member where

the RECEIVE command was issued. (For a file transmitted through RSCS, the actual origin node will be recorded.)

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, an entry is added in the NETLOG file. When an acknowledgment is received, it is also placed in this file.

```
File SMALL  DATA  A  sent   to SARAH  at NODE02 on 08/11/86 11:30:25
File SMALL  DATA  A  recv from SARAH  at NODE02 on 08/11/86 11:30:47 sent as SMALL DATA A
Ackn 08/11/86 11:30:47  recv  by SARAH  at NODE02 on 08/11/86 11:30:25
```

In the above example, the dates for the entries are in SHORTDATE date format.

Note: The date format for the date is the same as the first valid record in your *userid* NETLOG file. If it is a new file, the date format for the date will default to ISODATE (*yyyy-mm-dd*). To change the date formats in your *userid* NETLOG file, see "NETLCNVT" on page 566.

Here are examples of entries with FULLDATE and ISODATE date format:

```
File SMALL  DATA  A  sent   to SARAH  at NODE02 on 08/11/1986 11:30:25

File SMALL  DATA  A  sent   to SARAH  at NODE02 on 1986-08-11 11:30:25
```

In first example above, the user sent themselves a file called, SMALL DATA by using the SENDFILE command with the LOG and ACK options specified. The first line in the NETLOG file was placed in the file by the SENDFILE command. For more information, see "SENDFILE" on page 888.

The user then used the RECEIVE command (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 `recv` in this line means `received`. If a file is discarded (using DISCARD), the line contains `disc` instead of `recv`. (The file can be received with a different file ID than it was sent as.)

Then the user receives an acknowledgment. This shows whether the recipient received (`recv`) or discarded (`disc`) the file.

Responses

1. If you specify the FULLPROMPT or MINPROMPT option one of these prompts is displayed:

```
DMS1079R  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 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 DMS1079R, this message appears and you must enter a file ID of the form *fn* [*ft* [*fm*]].

```
DMS1080R  Enter the new name for fn ft fm
```

3. If you respond to prompt message DMS1080R with a file ID that names an existing file, you receive this prompt:

```
DMS1081R  Replace fn ft fm?
Reply 0 (NO), 1 (YES), or 2 (QUIT)
```

4. For spool files in the DISK DUMP or PUNCH format, RECEIVE issues the following response:

```
File fn1 ft1 fm1 received from userid at node
sent as fn2 ft2 fm2
```

   Additional responses may be issued by DISK LOAD or READCARD.

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 saved in a temporary file
     ```

     – If an error does not occur you receive this response so you are aware 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 copied to fn2 ft2 fm2 and fn1 ft1 fm1
then erased
```

7. Other responses include:

```
File fn ft has been discarded
Note fn ft has been discarded
Note fn ft added to fn NOTEBOOK fm
Ackn date time added to userid NETLOG
Ackn date time has been discarded
```

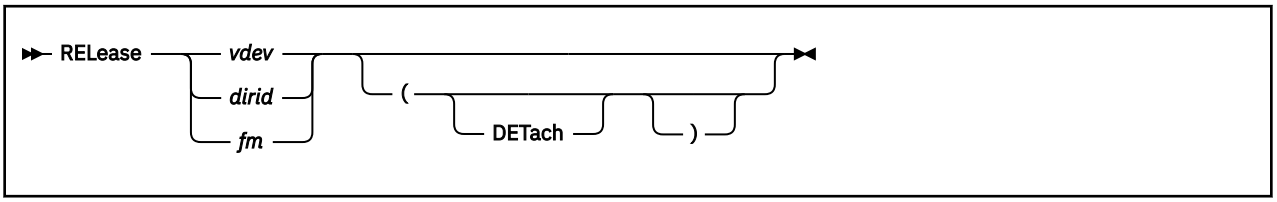
Messages and Return Codes

- DMS006E No read/write filemode accessed [RC=36]
- DMS024E File *fn* [*ft fm*] already exists[; specify REPLACE option] [RC=28]
- DMS037E Filemode *mode* is accessed as read/only [RC=36]
- DMS062E Invalid {character [*char*]|*} in [output] fileid [*fn ft [fm]*] [RC=20]
- DMS069E Filemode *mode* not accessed [RC=36]
- DSM630S Error accessing spool file [RC=36]
- DMS636W File *fileid* is empty; {minidisk|filepool *filepoolid*} does not support empty files [RC=74|88]
- DMS643E No class *class* files in your reader [RC=28]
- DMS644E All reader files are in HOLD status or not class *class* [RC=28]
- DMS653E Error executing *command*, rc=*rc* [RC=40]
- DMS655E Spoolid *nnnn* does not exist [RC=28]
- DMS671E Error receiving file *fn ft fm*; rc=*nn* from *command* [RC=100]
- DMS672E Virtual reader invalid or not defined [RC=36]
- DMS674E Reader is not ready [RC=36]
- DMS681E This is an unnamed file; specify filename and filetype [RC=88]
- DMS682E Error copying file *fn ft* A to {*fn ft fm*|*mode*}; rc=*nn* from COPYFILE [RC=100]
- DMS687E This is a {SYSTEM{HELD|DUMP|LOCK} file|file with a special format} and cannot be received| This file has a special format and cannot be received [RC=1|10]
- DMS743E File *fn ft fm* is in an invalid format [RC=40]
- DMS1123E Unknown response *text* ignored
- DMS1124W Spool file *spoolid* has been left in your reader because one or more files were not received [RC=1]
- DMS1138E Filesharing conflict [involving file *fn ft fm*] [RC=70]
- DMS3033W Spoolid *nnnn* contains an empty file; {minidisk|filepool} does not support empty files [RC=74|88]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411
Errors in the Shared File System	“File Pool Server Messages” on page 1414

RELEASE



Authorization

General User

Purpose

Use the RELEASE command to free an accessed disk previously accessed with the ACCESS command (“ACCESS” on page 30).

Operands

vdev
is the virtual device number of the disk to be released. The valid device numbers are X'0001' through X'FFFF'.

dirid
is the name of the directory to be released. For more information on the *dirid*, see “Naming Shared File System (SFS) Directories” on page 6.

fm
is the file mode letter for which the disk or directory is currently accessed.

Options

DETach
specifies the disk is to be detached from your virtual machine configuration; CMS calls the CP command DETACH.

Note: The DETACH option is ignored when you are releasing a directory.

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 is maintained in your storage. When a disk is released, this list of files is freed from your storage and that storage becomes available for other CMS commands and programs.

   However, when you release an SFS directory, you do not immediately recover storage. When CMS does keep the list of files in your storage, the space may not be freed until it is needed by a CMS command or a program.

   When you release a R/W 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; as a result, the file search time may be reduced for users who subsequently access the same disk.
4. When a disk or directory is released, its read-only extensions, if any, are not released. They may be referred to by their own file mode. If a disk or directory is then accessed with the original file mode

that they were extensions of, they revert to being extensions of 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 the disk was altered. The exception to this is when the ACCESS command has been issued with the ERASE option for a disk, and no files have been written to the disk since. RELEASE will not rewrite the directory in this case. For more information, see "Usage Notes" on page 815.
7. Detaching a minidisk (with a CP DETACH command) will cause CMS to implicitly release the disk if it is accessed. Also, if a minidisk address is redefined (with a CP DEFINE or REDEFINE command), CMS will implicitly release the disk if it is accessed.

Examples

If you want to release and detach the 498 disk accessed as your file mode B, issue:

```
release 498 (det
```

or

```
release b (det
```

If you want to release the directory .PROJECT1 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 DETACH option. It indicates the disk has been detached.

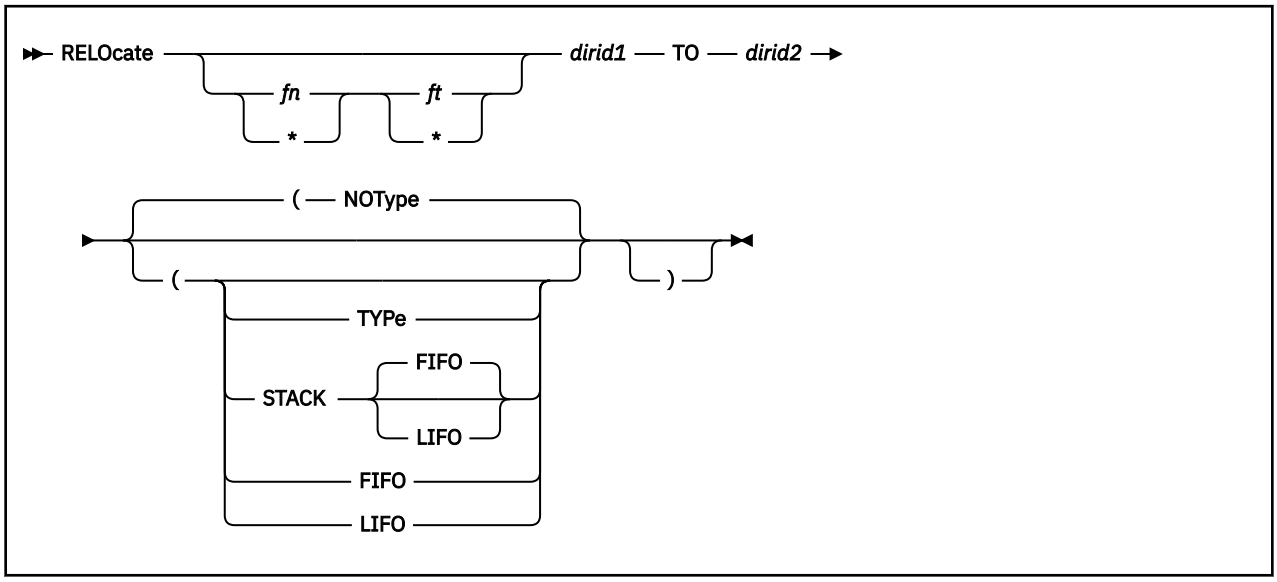
Messages and Return Codes

- DMS017E Invalid device address *vdev* [RC=24]
- DMS017E The CMS system disk cannot be released [RC=24]
- DMS028E No device specified [RC=24]
- DMS048E Invalid filemode *mode* [RC=24]
- DMS048E The CMS system disk cannot be released [RC=24]
- DMS065E *option* option specified twice [RC=24]
- DMS069E Filemode *mode* not accessed [RC=36]
- DMS069E Disk *vdev* not accessed [RC=36]
- DMS069E Directory *dirname* not accessed [RC=36]
- DMS109S Virtual storage capacity exceeded [RC=104]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411
Errors in the Shared File System	“File Pool Server Messages” on page 1414

RELOCATE



Authorization

General User

Purpose

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
- One of your directory structures to another directory you own

Operands

fn ft
is the name and type of the file to be moved. This parameter can be specified only for base files, aliases, and external objects that reside in a directory having the file control (FILECONTROL) attribute. You cannot specify *fn ft* for files that reside in a directory having the directory control (DIRCONTROL) attribute.

You can use special characters (* or %) to designate a group of files. For more information on these special characters, see "Using Pattern Matching to Specify Sets of Files" on page 14.

Not specifying *fn ft* indicates you are relocating a directory or directory structure.

dirid1
is either:

- The name of the directory that contains the files to be moved (if you specify *fn ft*)

  or

- The name of the parent directory (top node) of the directory structure to be moved (if you do not specify *fn ft*).

If you specify *fn ft*, you can specify *dirid1* as a file mode letter. Otherwise, it must be specified as a fully-qualified directory name. Your top directory cannot be moved. You must be the owner of *dirid1*. Both file control and directory control directories can be relocated.

When you relocate a directory structure, all subdirectories and files move with the directory structure.

For more information on *dirid*, see the "Naming Shared File System (SFS) Directories" on page 6.

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 can specify *dirid2* as a file mode letter if you choose. You must be the owner of *dirid2*. The target directory, *dirid2*, cannot be the same as *dirid1*. When *fn ft* is also specified, *dirid2* cannot be a directory control directory. 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. The default is NOTYPE.

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

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

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

Usage Notes

1. A relocated directory subtree keeps all its authorities and aliases. A relocated external object takes on the authorizations of the new parent directory. A relocated base file takes on the NEWREAD or NEWWRITE authorizations of the new directory.
2. When you move an alias, the alias remains in effect.
3. Erased and revoked aliases can be relocated.
4. Access to a directory is not broken by the RELOCATE command. A file control directory can be relocated while other users have the directory accessed.

   A directory control directory, on the other hand, cannot be relocated if it is accessed in read/write mode by anyone other than you; in that case, your attempt to relocate the directory will fail. You can relocate a directory control directory other users have accessed in read-only mode. They will not learn of the change until they reaccess the directory.
5. Any files you are moving must be closed.
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 locked by another user; and you cannot move a file that resides in a directory 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 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
```

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),

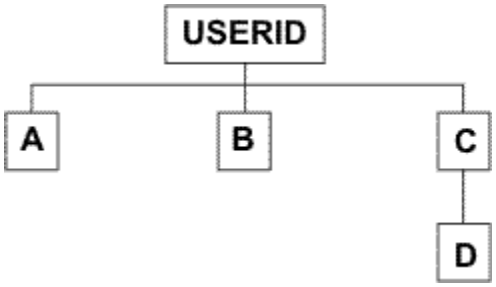
```
relocate .twolayer to .n1.n2.n3.n4.n5.n6.n7
```

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 or assembler program for an active file pool on the specified work unit, 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 REXX exec with ADDRESS COMMAND in effect
11. You can relocate a file in DFSMS/VM migrated status. The file data will not be recalled from the DFSMS/VM storage repository.
12. You cannot relocate files out of or into a directory control directory. You can, however, relocate a directory structure whose parent is a directory control directory. You can also relocate file control or directory control directories into a directory control directory. In either case, if you have the parent directory control directory accessed, you must have it accessed read/write.

Examples

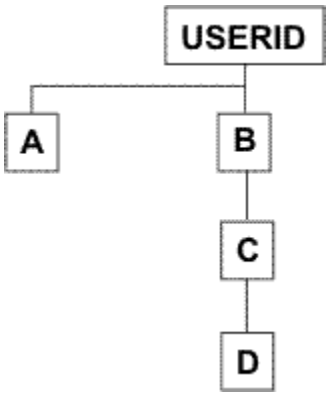
If you started with the following directory structure:



and issued,

```
relocate .c to .b
```

you would get this directory structure:



For more examples on using the RELOCATE command, see *z/VM: CMS User's Guide*.

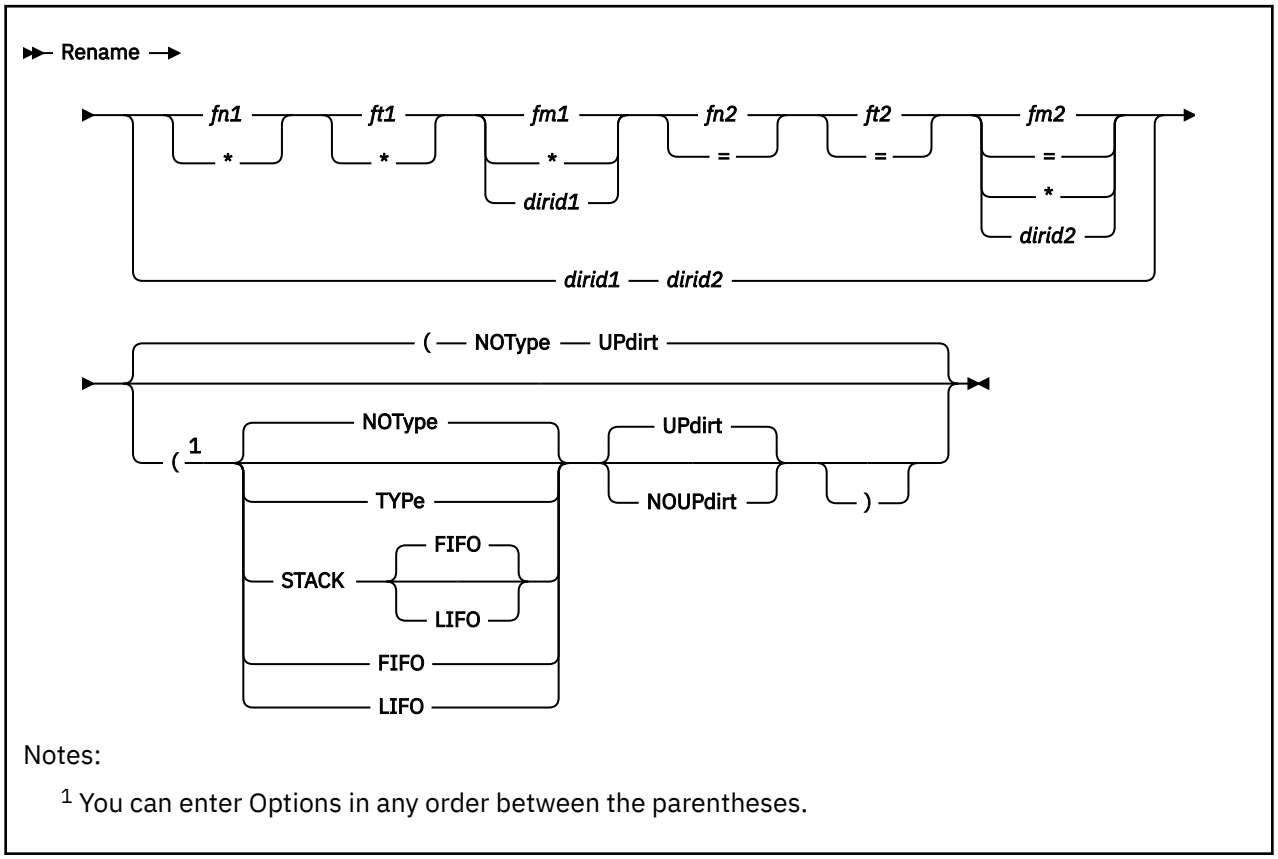
Messages and Return Codes

- DMS002E File *fn ft fm* | *dirname* not found [RC=28]
- DMS019E Identical fileids [RC=24]
- DMS065E *option* option specified twice [RC=24]
- DMS066E *option1* and *option2* are conflicting options [RC=24]
- DMS069E Filemode *fm* not accessed [RC=36]
- DMS105S Error *nn* writing file to XEDIT [RC=100]
- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS1131E Directory *dirname* already exists [RC=28]
- DMS1132E Invalid number of operands [RC=24]
- DMS1163E The RELOCATE command failed for *fn ft fm* | *dirname* [RC=*00*]
- DMS1187E Too many subdirectory levels in *dirid* [RC=24]
- DMS1188E Filemode *fm* is not associated with a directory [RC=74]
- DMS1189E Filemode *fm* is associated with a top directory [RC=24]
- DMS1223E There is no default file pool currently defined [RC=40]
- DMS1184E File *fn ft* or directory *dirname* not found or you are not authorized to use RELOCATE on one of these directories [RC=28]
- DMS1187E Too many subdirectory levels in *dirid* [RC=24]
- DMS1201E STACK option cannot follow FIFO or LIFO [RC=24]
- DMS1207E You cannot relocate a top directory [RC=88]
- DMS1208E Directory cannot be relocated within itself [RC=24]
- DMS1210E Directory *dirname* or directory *dirname* not found or you are not authorized to use RELOCATE on one of these directories. [RC=28]
- DMS1241E Directories specified are in different file pools [RC=88]
- DMS1251E Directories are from different directory structures [RC=88]
- DMS1290E File *fn ft fm* | *dirname* not relocated; source and target directories are the same [RC=24]
- DMS1291E There are no unused work units available. [RC=88]
- DMS2040E RELOCATE cannot be performed on a file in a directory control directory [RC=36]
- DMS2040E RELOCATE cannot be performed on a directory control directory that is accessed read-only [RC=36]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411
Errors in the Shared File System	“File Pool Server Messages” on page 1414
Errors in parsing a command	“Messages and Return Codes” on page 594

RENAME



Notes:

[1] You can enter Options in any order between the parentheses.

Authorization

General User

Purpose

Use the RENAME command to change:

- File IDs of one or more files on a minidisk or in a directory. The file can be a base file, an alias, or an external object.
- The name of an SFS directory you own.

Operands

fn1 ft1 fm1
fn1 ft1 dirid1
**** * ****

For a file on a minidisk this is the file name, file type, and file mode. For a file in an SFS directory, this could be the file name, file type, and a *dirid*. For more information on *dirid*, see "Naming Shared File System (SFS) Directories" on page 6.

You can use an asterisk (*) for any part of the file identifier to indicate any file that satisfies the other qualifications is renamed. Subdirectories that match the specified pattern are ignored and are not renamed.

fn2 ft2 fm2
fn2 ft2 dirid2
fn2 ft2 *
= = =
For a file on a minidisk, this is the file name, file type, and file mode. For a file in an SFS directory, this is the file name, file type, and a *dirid*.

You can use an equal sign (=) for any part of the file identifier to indicate the corresponding file identifier is unchanged. The file mode or *dirid* can also be specified as an asterisk (*), indicating the corresponding file mode or *dirid* is not changed.

dirid1
is the directory identifier of the SFS directory you want to rename. You can also use a file mode for the *dirid* if the directory is already accessed. For more information on *dirid*, see "Naming Shared File System (SFS) Directories" on page 6.

dirid2
is the new directory name. Variable *dirid2* cannot be expressed as a file mode when you rename a directory.

Options

Type
displays, at the terminal, the new identifiers of all the files being renamed. The file identifiers are displayed only when an asterisk (*) is specified for one or more of the file identifiers in the first file ID.

NOType
suppresses the display at the terminal of the new file identifiers of the files being renamed. The default is NOTYPE.

STACK FIFO
STACK LIFO
places the output 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 (*fn1, ft1, fm2,* or *dirid1*) in *fileid1*. The default is FIFO.

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

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

UPdirt
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. This is the default.

NOUPdirt
suppresses the updating of the master file directory when a file on a minidisk is renamed. For more information, see Usage Note "16" on page 825.

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 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, 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 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 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.
8. You cannot rename a file in an SFS directory if the file is open in read/write mode by any user or if you have it open in read-only mode. You can rename a file that is open by other users in read-only mode.

   If you specify an explicit file ID for renaming, the directory in which the file resides can be open or closed.
9. You cannot rename a file or subdirectory of a directory control directory while the parent directory is accessed read/write by another user or read-only by you. You can rename it while it is accessed read-only by other users.
10. You cannot rename a directory control directory if it is accessed read/write by anyone other than you. If you have it accessed, you must have it accessed in read/write mode.
11. For file control directories, QUERY SEARCH or QUERY ACCESSED will display the new directory name immediately. Users who have directory control directories accessed in read-only mode will have the old name returned until they release all accesses to the directory and then access it again.
12. When you rename a directory, all files in the directory must be closed, and the directory and all the subdirectories must be either closed or open with intent of FILE.
13. 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 you hold.
14. 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 you hold. These same rules apply for renaming a subdirectory in a locked directory.
15. Directories can be renamed only one level at a time. For example, changing

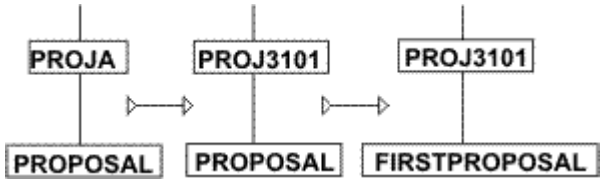
    .PROJA.PROPOSAL

    to

    .PROJ3101.FIRSTPROPOSAL

    would require two commands:

```
rename .proja .proj3101
rename .proj3101.proposal .proj3101.firstproposal
```



As illustrated by the first rename above, changing the name of a directory changes the names of all its subdirectories.

16. Usually, the master file directory 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.

17. You can only rename directories you own. You can rename another user's files if you are properly authorized. For files in file control directories, you must have write authority to the file and write authority to the directory that contains the file. For files in directory control directories, you must have directory control write (DIRWRITE) authority to the directory that contains the file.

    You can also rename another user's alias. All aliases exist in file control directories. To rename an alias, you need read authority to the base file and write authority to the directory containing the alias.

    You need write authority to the directory to rename an external object. You do not have to own the directory.

18. To use the fm form of *dirid* when renaming a file, the directory must be accessed in read/write mode. Other users' directories, by default, are accessed in read-only mode. To access another user's directory in read/write mode, use the FORCERW option on the ACCESS command.

19. You can issue RENAME from the command (Cmd) column on any of the FILELIST screens. For example, to rename a file in a directory accessed in read/write mode, enter:

    ```
    rename / newname = =
    ```

    If you are authorized to rename the file, but the directory is accessed in read-only status, enter:

    ```
    rename /ntd newname = =
    ```

    This indicates you choose to rename the *fn ft* directory to a new file name and keep the same file type and directory name as displayed.

    By default, directories owned by other users are accessed in read-only status. To access another user's directory in read/write status, use the FORCERW option on the ACCESS command.

    For more examples on using the RENAME command, see *z/VM: CMS User's Guide*.

20. 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.

21. The recoverability and overwrite attributes of the file are unaffected by the RENAME command, regardless of the new file mode number assigned.

22. You can rename a file that has been placed in DFSMS/VM migrated status. The file data will not be recalled from the DFSMS/VM storage repository.

23. RENAME can be used on mixed case file IDs when FILELIST is entered with the MIXEDON option.

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.

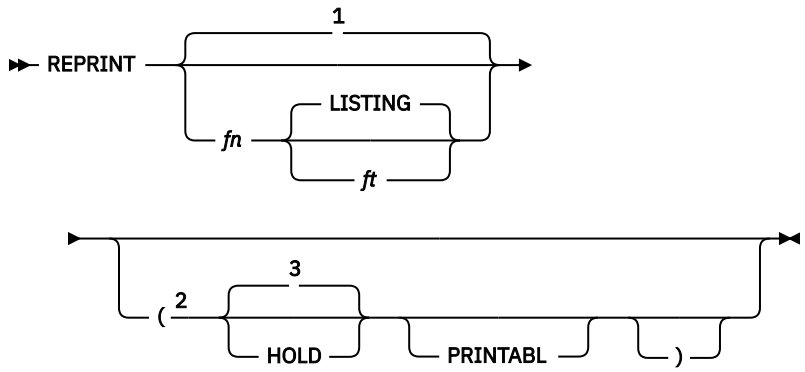
Messages and Return Codes

- DMS002E File *fn ft fm* | *dirname* not found [RC=28]
- DMS019E Identical fileids [RC=24]
- DMS030E File *fn ft fm* already active [RC=28]
- DMS037E Filemode *mode* is accessed as read/only [RC=36]
- DMS048E Invalid filemode *mode* [RC=24]
- DMS051E Invalid mode change [RC=24]
- DMS054E Incomplete fileid specified [RC=24]
- DMS069E Filemode *mode* not accessed [RC=36]
- DMS1131E Directory *dirname* already exists [RC=28]
- DMS1184E File *fn ft fm* | *dirname* not found or you are not authorized for it [RC=28]
- DMS1184E Directory *dirname* not found or you are not authorized for it [RC=28]
- DMS1199E You cannot rename a top directory [RC=24]
- DMS1226E Invalid subdirectory name change. Only the last qualifier of the specified subdirectory can be renamed. [RC=24]
- DMS1241E Directories specified are in different file pools [RC=88]
- DMS1257E The RENAME command is invalid on a file in a directory that you do not own [RC=76]
- DMS1257E The RENAME command is invalid on a directory that you do not own [RC=28]
- DMS1308E The filemode number of an alias must be the same as the filemode number of the base file [RC=24]
- DMS1309I Command completed successfully, but the filemode number of the alias is the same as the filemode number of the base file [RC=0]
- DMS2040E RENAME cannot be performed on a directory control directory that is accessed read-only [RC=36]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411
Errors in the Shared File System	“File Pool Server Messages” on page 1414

REPRINT



Notes:

[1] The default is the same file name and file type as the reader spool file.
[2] You can enter Options in any order between the parentheses.
[3] The default is to purge the file.

Authorization

General User

Purpose

Use the REPRINT command to transfer printer files from the virtual reader to the virtual printer without losing carriage control characters. Any tag information associated with the file is displayed when the file is processed.

Operands

fn
allows a different file name to be assigned to the printer spool file. The default is to assign the same file name and file type as the reader spool file. If a file name is specified without a file type, the default file type is LISTING.

ft
allows a different file type to be assigned to the printer spool file. The default file type if a file name is specified is LISTING. If neither file name nor file type is specified, the default is to assign the same file name and file type as the reader spool file.

HOLD
causes the input spool file to be held in the virtual reader. The default is to purge the file. This option is automatically assumed if any errors occur during execution of the command.

PRINTABL
forces the translation of the input file data to the standard 64-character set. Unprintable characters are changed to blanks (X'40').

Usage Notes

1. REPRINT will not transfer reader files in HOLD status. You should verify the correct files get transferred. REPRINT transfers the first printer file in your virtual reader not in HOLD status. If REPRINT does not find any files it can process in your virtual reader, you will receive a return code 4 (no input print file).

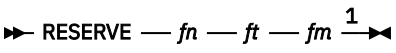
2. REPRINT does provide support for 3800 printer files, but it does not provide support for table reference characters (TRCs) in those files.
3. You can also use the preferred TRANSFER command to transfer your closed spool files to a specified user or queue, or to reclaim closed spool files you created. For details, see *z/VM: CP Programming Services*.
4. When REPRINT is issued from the RDRLIST screen, the symbol, /o, must prefix the REPRINT command string. For example:

```
/o REPRINT fn ft
```

Messages and Return Codes

- DMS113S Printer 00E not attached [RC=6]
- DMS205E No files in your reader [RC=4]
- DMS2166E Invalid channel command code found [RC=8]
- DMS2167I The tag for printer file number *spoolid* was:
- DMS2181E Unknown printer type. [RC=2]
- DMS2182E File LRECL too big for defined printer. [RC=10]
- DMS2183I *num* lines with unprintable characters were found [RC=0]

RESERVE



Notes:

[1] If a file mode number is not specified the default is 6.

Authorization

General User

Purpose

Use the RESERVE command to allocate all available blocks of a formatted CMS minidisk to a unique CMS file.

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), the file mode number defaults to 6.

Format of the File

When the RESERVE command completes, the defined files have this format:

- File name, file type, and file mode letter as defined by the user.
- File mode number 6 (indicating "overwrite") 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. For more information on the RESERVE command using the DISKID function, see *z/VM: CMS Macros and Functions Reference*.

  The file that is created can be read or written through 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:

Start Block	No. of Blocks	Content
1	2	IPL record
3	1	Volume label
4	2	CMS file directory

Start Block	No. of Blocks	Content
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

Examples

Suppose you have a disk device with one cylinder formatted with 4096-byte block size. There are 180 blocks available. After you issue the RESERVE command, the file that is created has this format:

```
Blocks used by CMS file system |     data blocks
 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |10 |11 |...|178|179|180
```

Where:

Physical Block Number	Contains
1 and 2	IPL records
3	volume label
4 or 5	CMS directory file
6	allocation map
7	alternate allocation map
8	CMS level 1 pointer block
9–180	data blocks 1 through 172

The data blocks of the file created are now organized *sequentially* (blocks 9-172) 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 172) is actually physical block 180.

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 the 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 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 Enter, 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

- DMS037E Filemode *mode* (*vdev*) is accessed as read/only [RC=36]
- DMS042E No fileid(s) specified [RC=24]
- DMS054E Incomplete fileid specified [RC=24]
- DMS069E Filemode *mode* not accessed [RC=36]
- DMS069E Output filemode *mode* not accessed [RC=36]
- DMS070E Invalid parameter *parameter* [RC=24]
- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS113S Disk(*vdev*) not attached [RC=100]
- DMS223E No filemode specified [RC=24]
- DMS260E Disk not properly formatted for RESERVE [RC=16]
- DMS908E File system error detected at virtual address *vdev*; reason code *nn* [RC=100]
- DMS909E Permanent I/O error on *vdev*; csw=*csw*, sense=*sense* [RC=100]
- DMS1264E 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 their location are:

Reason	Location
Errors in command syntax	"Command Syntax Error Messages" on page 1411

RETURN

Authorization

General User

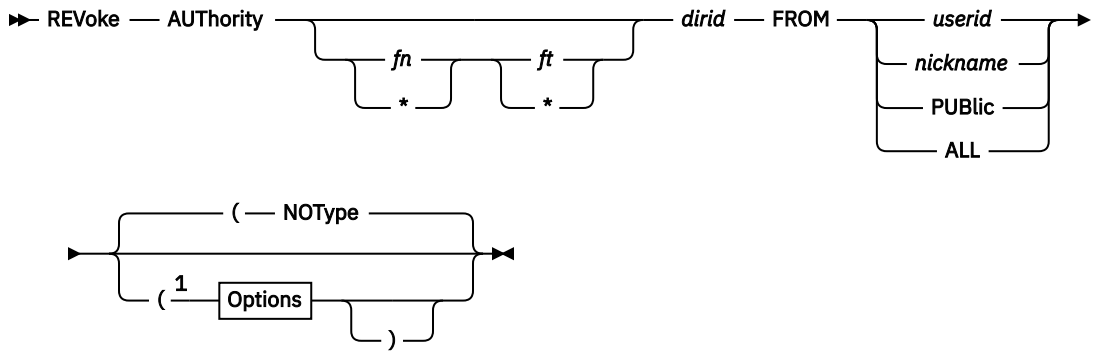
Purpose

Use the RETURN command to return to edit mode after the CMS subset environment has been entered from XEDIT through the CMS subcommand. For more information, see the CMS subset mode and the CMS subcommand, see *z/VM: CMS User's Guide* and *z/VM: XEDIT Commands and Macros Reference*.

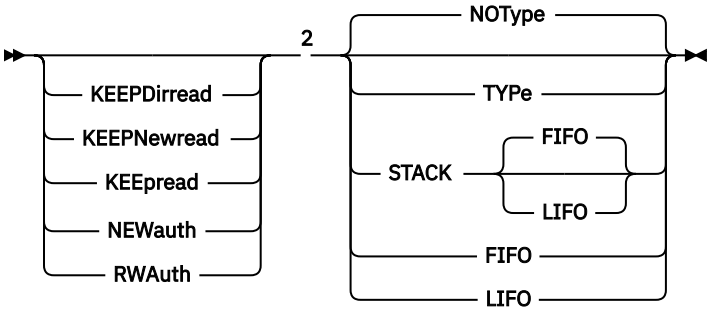
Usage Notes

The user can return to XEDIT by entering RETURN on the command line.

REVOKE AUTHORITY



Options



Notes:

[1] You can enter Options in any order between the parentheses.
[2] For a file control directory, some option combinations are available: NEWAUTH KEEPREAD, RWAUTH KEEPNEWREAD, KEEPREAD KEEPNEWREAD, and RWAUTH NEWAUTH (default).

Authorization

General User

Purpose

Use the REVOKE AUTHORITY command to remove authorities you have granted to others.

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. For more information on pattern matching using these characters, see "Using Pattern Matching to Specify Sets of Files" on page 14.

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. You can also use a file mode for the *dirid* if the directory is already accessed. For more information on *dirid*, see "Naming Shared File System (SFS) Directories" on page 6.

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 granted individually.

FROM ALL
revokes authority from all users for a file or directory. If PUBLIC authority was granted, it is also revoked.

Options

KEEPDirread
specify this option when you want to change a user's authority from DIRWRITE to DIRREAD. KEEPDIRREAD is valid only when a directory is identified (omit the file name and file type) and when the directory has the directory control (DIRCONTROL) attribute.

KEEPNewread
specify this option when you want to change a user's authority from NEWWRITE to NEWREAD. KEEPNEWREAD is valid only when a directory is identified (without a file name and file type) and when that directory has the file control (FILECONTROL) attribute.

KEEpread
specify this option when you want to change a user's authority from WRITE to READ. This option is valid for file control directories and for files within file control directories. When specified for a file, aliases the user has to the file are kept in effect.

NEWauth
specify this option when you want to remove both NEWREAD and NEWWRITE authority from a specified user. NEWAUTH is valid only when a directory is identified (without a file name and file type) and when that directory has the file control (FILECONTROL) attribute.

RWAuth
specify this option when you want to remove both READ and WRITE authority from a specified user. RWAUTH is valid only for file control directories or files within file control directories.

TYPe
displays information at the terminal.

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

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

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

LIFO
specifies the output 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. The authorities you can grant and revoke are:

- For SFS file control directories:
  - READ authority
  - WRITE authority (which implies READ authority)
  - NEWREAD authority
  - NEWWRITE authority (which implies NEWREAD authority)

    Note: NEWREAD and NEWWRITE authority are independent from READ or WRITE authority.
- For files within file control directories:
  - READ authority
  - WRITE authority (which implies READ authority)
- For SFS directory control directories:
  - DIRREAD authority
  - DIRWRITE authority (which implies DIRREAD authority)
- For files within directory control directories:

  Not applicable. You cannot grant or revoke authorities on individual files within directory control directories. The ability to read from or write to files in directory control directories is derived from DIRWRITE and DIRREAD authority on the directory in which the file resides.
- For external objects:

  Not applicable. You cannot revoke authority to external objects. An external object has no authorities pertaining to it. The remote name in the external object may be queried by anyone with read authority to the parent directory.

3. To revoke all authorizations for an object, enter a REVOKE command *without* specifying any authority options. Suppose you have a directory named .VERSION1. To revoke all authorizations user Barb has on the directory, enter:

   ```
   revoke authority .version1 from barb
   ```

   If .VERSION1 is a file control directory, Barb loses any READ, WRITE, NEWREAD, or NEWWRITE authority she might have. If .VERSION1 is a directory control directory, Barb loses either DIRREAD or DIRWRITE authority (whichever she has).

   Now suppose you want to revoke all authority Barb has on a file named EXPENSE DATA in the .FINANCE directory. (The .FINANCE directory would have to be a file control directory because authorities cannot be granted on individual files within directory control directories.) You would enter:

   ```
   revoke authority expense data .finance from barb
   ```

   Barb loses READ and WRITE authority to the file. Another way to revoke both READ and WRITE authority from a file in a file control directory is by specifying the RWAUTH option.

   For more information on the REVOKE AUTHORITY command, see *z/VM: CMS User's Guide*.
4. To downgrade a user's authorization, specify one of the authority options. When you specify an authority option, only the indicated authority is changed. Other authorizations are retained.
5. For file control directories, you can specify two authority options in a single command. The allowed combinations are:
   - NEWAUTH KEEPREAD
   - RWAUTH KEEPNEWREAD
   - KEEPREAD KEEPNEWREAD
   - RWAUTH NEWAUTH (default)
6. Revoking authority through an alias revokes the authority on the base file.

7. You can revoke authority on a file or directory 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.
8. When you revoke authority from a file in a file control directory and that file is open, the authority is revoked after the file is closed. To revoke authority from a file control directory, all files within the directory must first be closed.

   Individual file authorizations cannot be granted or revoked for directory control directories. The only authorities related to directory control directories are directory control read (DIRREAD) and directory control write (DIRWRITE) authority. These authorities are granted on the directory, not the files within it. When you enter a REVOKE AUTHORITY command for a directory control directory, the command will fail if any file within the directory is open for writing.
9. When you revoke authority from an accessed directory, the directory is released from the affected user and no message is issued.
10. You cannot revoke authority for a directory control directory if it is accessed in read/write mode by anyone other than you. Any user who has the directory accessed in read-only mode when his authority is revoked will lose authority when he releases the directory.
11. If the REVOKE AUTHORITY command fails for a file specified by special characters, the processing continues with the next file name that matches the pattern.
12. If you choose to revoke authority from 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.
13. The REVOKE AUTHORITY command can only revoke authorities established through the GRANT AUTHORITY command or routine. For any authorities granted through the External Security Manager (ESM), an ESM command must be used to revoke the authority.
14. You can invoke the REVOKE AUTHORITY command from the command line, from an exec, or as a function from a program. No error messages are issued if REVOKE AUTHORITY 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 REXX exec with ADDRESS COMMAND in effect

Messages and Return Codes

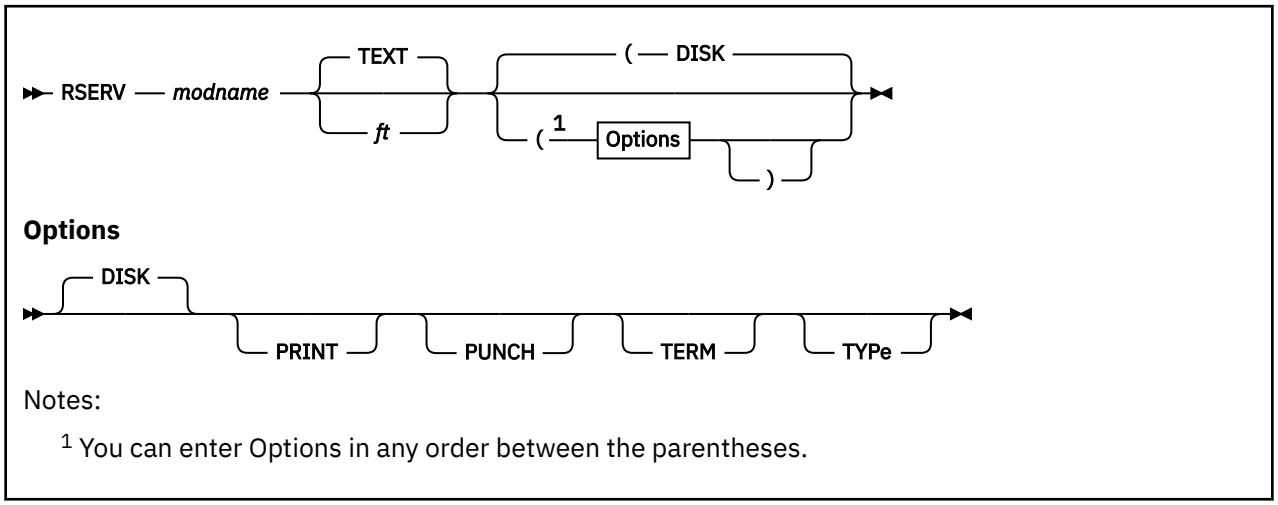
- DMS002E File *fn ft fm* | *dirname* not found [RC=28]
- DMS065E *option* option specified twice [RC=24]
- DMS066E *option1* and *option2* are conflicting options [RC=24]
- DMS069E Filemode *fm* not accessed [RC=36]
- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS149E Userid *userid* not valid [RC=32]
- DMS637E Missing nodeid for the AT operand [RC=24]
- DMS647E Userid not specified for *nickname* in *userid* NAMES file [RC=32]
- DMS647E Localid not specified for *userid* at *node* in *userid* NAMES file [RC=32]
- DMS653E Error executing *command* [RC=40]
- DMS1163E The REVOKE AUTHORITY command failed for *fn ft fm* | *dirname* [RC=0]
- DMS1184E File *fn ft fm* | *dirname* not found or you are not authorized for it [RC=28]
- DMS1184E Directory *dirname* not found or you are not authorized for it [RC=28]
- DMS1187E Too many subdirectory levels in *dirid* [RC=24]
- DMS1188E Filemode *fm* is not associated with a directory [RC=74]
- DMS1189E Filemode *fm* is associated with a top directory [RC=24]

- DMS1201E STACK option cannot follow FIFO or LIFO [RC=24]
- DMS1223E There is no default file pool currently defined [RC=40]
- DMS1245W Because *userid* owns *fn ft fm*|*dirname*, the authority cannot be revoked [RC=4]
- DMS1247W Public [READ | WRITE | NEWREAD | NEWWRITE | DIRREAD | DIRWRITE] authority did not previously exist on *fn ft fm*| *dirname* [RC=4]
- DMS1247W No users had [READ | WRITE | NEWREAD | NEWWRITE | DIRREAD | DIRWRITE] authority to *fn ft fm*|*dirname* [RC=4]
- DMS1248W Specified authorization revoked, but external security is still in effect for *fn ft fm*|*dirname* [RC=4]
- DMS1287W You do not own file *fn ft fm*|*dirname* [RC=4]
- DMS2039E KEEPDIRREAD | KEEPNEWREAD | NEWAUTH option cannot be specified on a file [RC=24]
- DMS2039E KEEPDIRREAD option cannot be specified for a file control directory [RC=24]
- DMS2039E KEEPNEWREAD | KEEPREAD |NEWAUTH option cannot be specified for a directory control directory [RC=24]
- DMS2039E KEEPDIRREAD| KEEPNEWREAD | NEWAUTH option cannot be specified with the current level of file pool *filepoolid* [RC=88]
- DMS2040E REVOKE AUTHORITY cannot be performed on a file in a directory control directory [RC=24]
- DMS2040E REVOKE AUTHORITY cannot be performed on a directory control directory that is accessed read-only [RC=36]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411
Errors in the Shared File System	“File Pool Server Messages” on page 1414

RSERV



Notes:

[1] You can enter Options in any order between the parentheses.

Authorization

General User

Purpose

Use the RSERV command in CMS/DOS to copy, display, print, or punch a VSE relocatable module from a private or system library.

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. If a file type is not specified, *ft* defaults to TEXT. The file name is always the same as the module name.

Options

You may specify as many options as you choose on the RSERV command, depending on which functions you want to perform. If you specify more than one option, CMS uses the last option specified.

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.

TYPe
displays the relocatable module at your terminal. (This option has the same function as the TERM option.)

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.

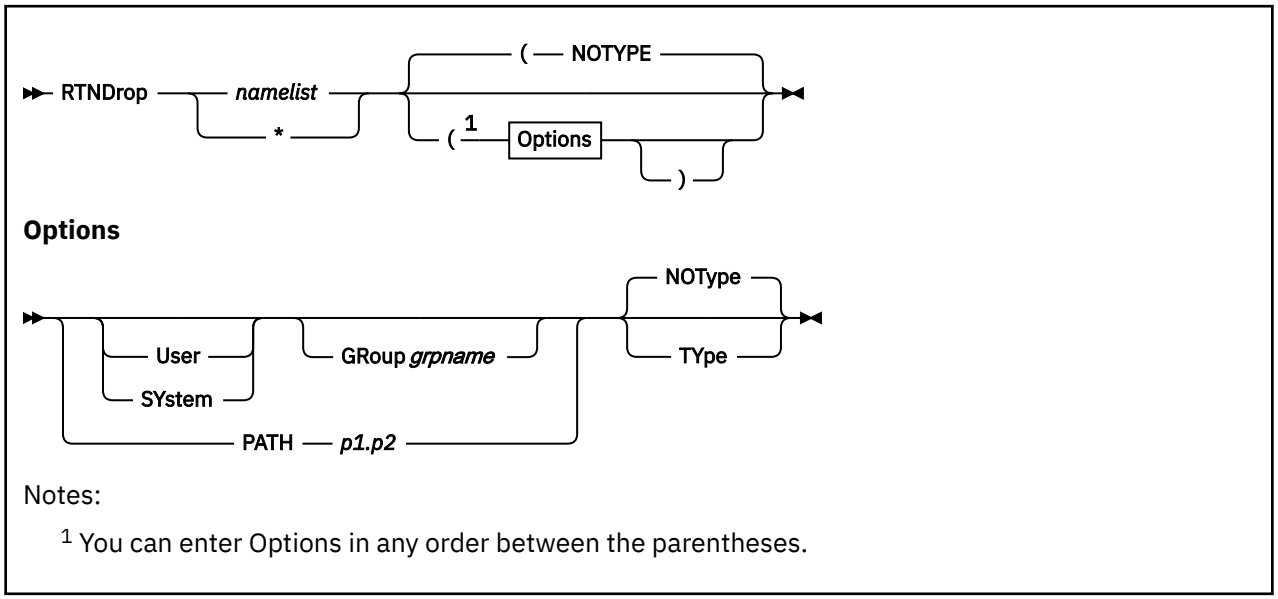
Responses

If you use the TERM option, the relocatable module is displayed at the terminal.

Messages and Return Codes

- DMS003E Invalid option: *option* [RC=24]
- DMS004E Module *module* not found [RC=28]
- DMS006E No read/write A filemode accessed [RC=36]
- DMS070E Invalid parameter *parameter* [RC=24]
- DMS089E Open error code *nn* on SYSRLB [RC=36]
- DMS097E No SYSRES volume active [RC=36]
- DMS098E No module name specified [RC=24]
- DMS099E CMS/DOS environment not active [RC=40]
- DMS105S Error *nn* writing file *fn ft fm* on disk or directory [RC=100]
- DMS113S Disk(*vdev*) not attached [RC=100]
- DMS411S Input error code *nn* on SYS*aaa* [RC=*rc*]

RTNDROP



Notes:

[1] You can enter Options in any order between the parentheses.

Authorization

General User

Purpose

Use the RTNDROP command to undo the binding of callable services library (CSL) routines that were loaded by the RTNLOAD command.

Operands

namelist
specifies the list of CSL routines 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, GROUP, or PATH option is specified, all routine versions associated with each run name satisfying the specified criteria are dropped. If none of the three options is specified, only the most recently loaded routine version associated with each run name is dropped.

*****
specifies all routines satisfying the specified options (USER, SYSTEM, GROUP, and PATH) are to be dropped. If none of these four options is specified, an error message is generated.

Options

User
specifies only routines loaded with the USER attribute are to be dropped.

SYstem
specifies only routines loaded with the SYSTEM attribute are to be dropped.

GRoup *grpname*
specifies only routines loaded with the given *grpname* are to be dropped.

PATH *path*
specifies only routines with the given path(s) are to be dropped. *Path* can be *p1.p2* or '*', where '*' specifies only routines loaded with a path (the path value makes no difference in this case) are to be dropped.

Values for *p1* can be:

1 - 1024
specifying a particular path

*****
specifying all values between 1 and 1024

Values for *p2* can be:

1 - 250
specifying a particular path starting with *p1*

*****
specifying all paths starting with *p1*

TYpe
indicates an informational message is to be issued for each routine that is successfully dropped.

NOType
indicates 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
   - CSLMAP – displays information about currently loaded CSL routines
   - RTNLOAD – loads a CSL routine
   - RTNMAP – lists the CSL routines currently loaded
   - RTNSTATE – determines the status of a specific CSL routine
4. An entire library subgroup can be dropped by specifying the subgroup name as the GROUP option's *grpname*. The subgroup must not have been previously loaded using the RTNLOAD command with the NOGRROUP or GROUP options specified.
5. A protected routine is a routine residing within a segment CSL library which, after the initial RTNLOAD, cannot be subsequently replaced by performing additional RTNLOAD or RTNDROP commands.
6. RTNDROP cannot remove a protected routine that has been loaded. Purging the segment is required.
7. RTNDROP cannot remove a drop-protected routine that has been loaded. However, a drop-protected routine can be replaced as the active version by using RTNLOAD to load a replacement.
8. You can determine what protected and drop-protected routines are currently in use by using the CSLMAP command. For more information, see “CSLMAP” on page 128.

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.

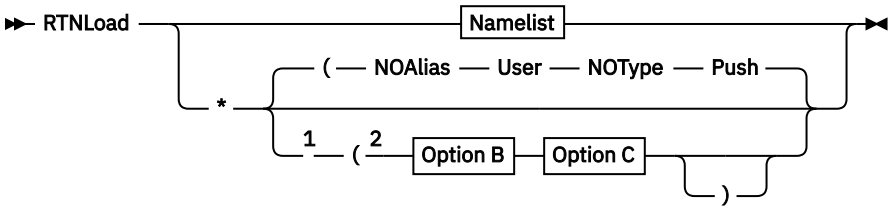
Messages and Return Codes

- DMS065E *option* option specified twice [RC=24]
- DMS066E *opt1* and *opt2* are conflicting options [RC=24]
- DMS624W No CSL routines are loaded [RC=28]
- DMS1086E Namelist is invalid: "*" is not valid with routine names [RC=24]
- DMS1087E Either the USER, SYSTEM, GROUP, or PATH option must be specified if namelist is specified as "*" [RC=24]
- DMS1088W Routine *rtnname* cannot be dropped because it is not loaded [RC=4]
- DMS1088W Routine *rtnname* cannot be dropped because it was not loaded with the specified attribute [RC=4]
- DMS1088W Routine *rtnname* cannot be dropped because it was not loaded with the specified group name [RC=4]
- DMS1088W One or more versions of routine *rtnname* matched the requirements but were protected and were not dropped [RC=4]
- DMS1088W Routine *rtnname* cannot be dropped because it was not loaded with the specified CSL path [RC=4]
- DMS1089I *rtnname* has been dropped
- DMS1090E Invalid CSL path *path* specified [RC=24]
- DMS1129W No routines were dropped. Either none of the routines matched the requirements or all the routines matching the requirements were protected [RC=4]
- DMS1129W Some of the routines matching the requirements were protected and were not dropped [RC=4]

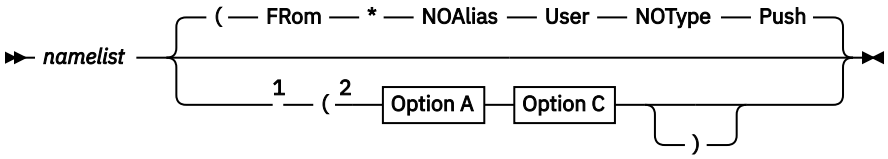
Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

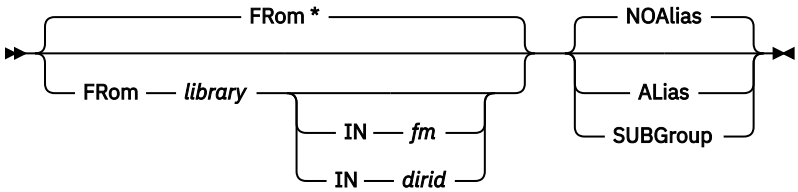
RTNLOAD



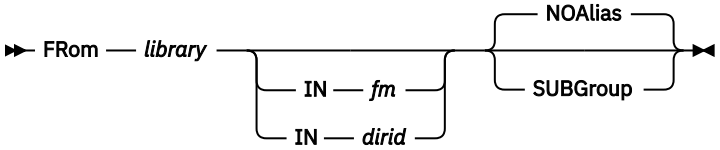
Namelist



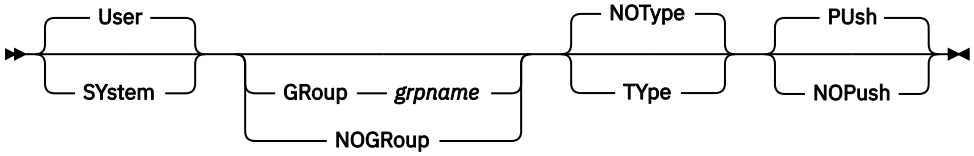
Option A



Option B



Option C



Notes:

[1] You can enter Options in any order between the parentheses.
[2] The default options are shown above the line in the options groups.

Authorization

General User

Purpose

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, until that version is dropped or a new version is loaded.

The routines reside in a DASD-based callable service library, a saved segment-based callable services library, or a CMS nucleus-resident callable services library.

Operands

namelist
specifies a list of library subgroup names or a list of routine names to be searched for, loaded into user storage if necessary, and bound. The following is what is included in the lists depending on the option specified:

NOAlias
Individual routine names

ALias
Routine name/alias name pairs

SUBGroup
Library subgroup names

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
rtnload subgr1 subgr2 subgr3 ... (SUBGroup
```

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 an alias given when the routine was loaded.

*****
when used with the SUBGROUP option, specifies all routines with subgroup names from the specified library are to be loaded. When used with the NOALIAS option, this specifies 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.

Note:

1. Saved segment resident libraries are searched first.
2. CMS nucleus-resident libraries are searched second.
3. Minidisk-resident and file pool server directory-resident libraries are searched last.
4. Libraries that can no longer be found are ignored.

FROM *
specifies libraries should be searched using the library search order; the default callable services libraries, VMLIB and VMMTLIB, will be implicitly appended to the current search order. For more information, see “GLOBAL” on page 363. 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. For more information on how to specify *dirid*, see “Naming Shared File System (SFS) Directories” on page 6.

ALias
specifies 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 routines are loaded under their original names. This is the default.

SUBGroup
indicates the names in *namelist* represent subgroup names. All routines with the specified subgroup name are loaded. The subgroup name becomes the group name unless a different name is specified with the GRoup *grpname* option.

User
specifies the routines are to be dropped if an abend occurs. This is the default.

SYstem
specifies these routines are to be retained if an abend occurs.

Note: A saved segment resident routine loaded with this option will not survive abend processing unless the saved segment has been loaded with the SYSTEM option on the SEGMENT LOAD command.

GRoup *grpname*
identifies the name of a collection of related routines. *Grpname* replaces any subgroup names specified for the routines loaded.

NOGRoup
specifies no group name is to be saved for the routines loaded.

TYpe
indicates an informational message is to be issued for each routine successfully loaded.

NOType
indicates messages are not to be issued when routines are loaded. This is the default.

PUSH
specifies that if a routine with 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 when using the DMSCSL or CSLFPI interface. A RTNDROP of this run name drops the most recently loaded version and restores the most recently saved version as the current version. If a path is used by the new routine version, the path will be activated for the routine if it is not already active. If the path is already active it will be updated for the new version. Any other paths activated for the name are unaffected, giving access to previously loaded versions. PUSH is the default. If the routine is not already loaded, this option is ignored.

Note: PUSH fails when trying to RTNLOAD a new version under a run name that is protected.

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 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
   - CSLMAP – displays information about currently loaded CSL routines
   - 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.
9. If you do not specify either the GROUP or NOGROUP options, the subgroup name assigned to the routine when the library was built is saved as the group name.
10. If you RTNLOAD direct call routines, specifying the 1st path number between 513 and 1024, the path will be locked to the run name specified. You cannot RTNLOAD using the same path and a different run name until all routine versions using the first run name and the path in contention are dropped.
11. You cannot directly call a routine version if the version does not use a path.
12. A protected routine is a routine residing within a DCSS CSL library which, after initial RTNLOADing, cannot be subsequently replaced by performing additional RTNLOADs or RTNDROPs. The routine can only be removed by purging the segment it resides in.
13. RTNLOAD cannot change a protected routine that has been loaded until the DCSS is purged.
14. A drop-protected routine is a routine that cannot be dropped once it is loaded. However, unlike a protected routine, it can be replaced as the currently active version by using RTNLOAD.
15. You can see which routines are protected and which protection they have by using either the CSLLIST or CSLMAP command. In the Protect column of the Attribute Display Screen, 'X' is displayed if the routine is protected against dropping and replacement, 'D' is displayed if the routine is drop-protected, and '-' is displayed if it has no protection.
16. If you use RTNLOAD with the ALIAS option to load a directly-callable routine, the path assigned to the run name is the same as the path assigned to the alias name. If you use RTNLOAD with the ALIAS option to load a routine not directly-callable, any path associated with the alias name is ignored, and no path is assigned.
17. Some routines, such as OpenExtensions routines, cannot be loaded with an alias. Unlike most CSL routines, these routines do not carry the return code as the first parameter in the parameter list. The direct call interface requires the call routing code segment (stub), which calls the routine, must carry return code position information in case an interface error prevents completion of the call. An alias stub cannot provide this information.

    You can see which routines this note applies to by using either the CSLLIST or CSLMAP command. In the Interface column of the Attribute Display Screen, '2' is displayed if the routine cannot be loaded using an alias name.

Examples

1. `rtnload * (from mylib`

   loads all the routines from the callable services library called MYLIB. RTNLOAD first searches saved segments to find MYLIB, and if unsuccessful, follows the CMS search order.
2. `rtnload sample xyzzy (from mylib alias`

indicates the routine SAMPLE is to be loaded from the callable services library MYLIB, and 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.

6. `rtnload math history (from mylib subgroup`

   loads all routines belonging to subgroups MATH and HISTORY from the library MYLIB.

7. If you RTNLOAD a directly callable routine and specify the ALIAS option, the path assigned to the run name is the same as the path assigned to the alias name. If you RTNLOAD a routine not directly callable, any path associated with the alias name is ignored. No path is assigned.

Messages and Return Codes

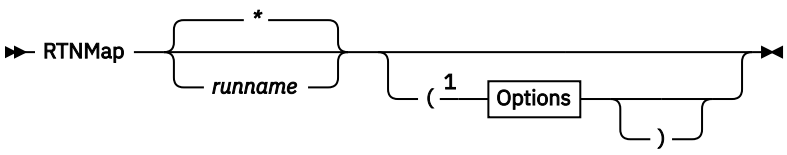
- DMS065E *option* option specified twice [RC=24]
- DMS066E *opt1* and *opt2* are conflicting options [RC=24]
- DMS639E Error in *rtnname* routine; return code was *retcode*
- DMS1084E ALIAS option is not valid if namelist is specified as "*" [RC=24]
- DMS1084E ALIAS option is not valid if namelist has an odd number of entries [RC=24]
- DMS1085E A library name must be specified if namelist is specified as "*" [RC=24]
- DMS1085E A library name must be specified if the IN option is specified [RC=24]
- DMS1086E Namelist is invalid: "*" is not valid with routine names [RC=24]
- DMS1086E Namelist is invalid: more than 256 names are specified [RC=24]
- DMS1089I *rtnname* has been loaded [RC=0]
- DMS1089I *rtnname* cannot be loaded using an Alias name. [RC=8]
- DMS1090E Invalid routine name *rtname* specified [RC=24]
- DMS1097E Routine *rtnname* not found [RC=8]
- DMS1097E Subgroup *subgroup name* not found [RC=8]
- DMS1097E Routine *rtnname* could not be found as a ROUTINE entry in a library. [RC=8]
- DMS1098E Some of the specified routines were not found. These names could not be found as ROUTINE entries in a library. [RC=8]
- DMS1098E None of the specified subgroups were found [RC=8]
- DMS1098E None of the specified routines were found [RC=8]
- DMS1099W *rtnname* has already been loaded [RC=4]
- DMS1100E No filemode is available to access *dirid* [RC=12]
- DMS1136E Unable to gain access to library *libname* [RC=28]
- DMS1136W Unable to gain access to library *libname* [RC=4]
- DMS2502E Routine *rtnname* not loaded. Unique path *path* locked for another routine. [RC=4]
- DMS2502E Routine *rtnname* not loaded. Current version is protected. [RC=4]
- DMS2502E Routine *rtnname* not loaded. Routine cannot have an alias. [RC=4]
- DMS2503S Unable to initialize internal tables [RC=1,2,3,9]

- DMS2503S Unable to initialize CSL environment [RC=1,2,3,9]

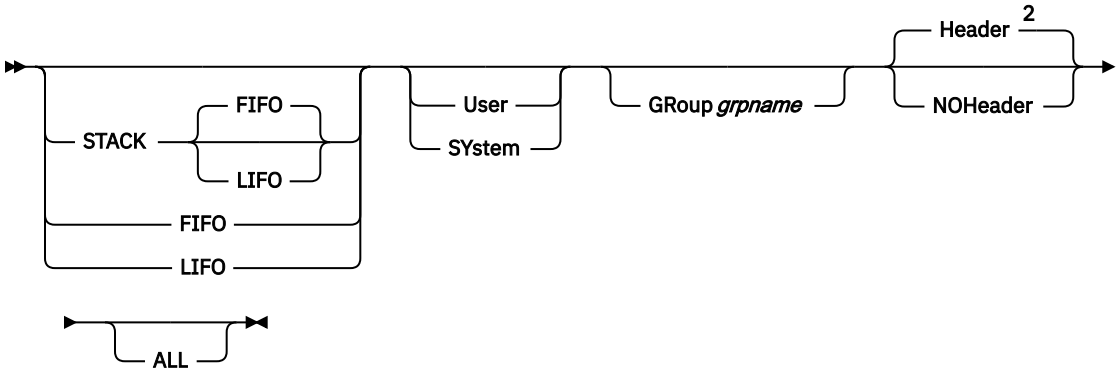
Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411
Errors in the Shared File System	“File Pool Server Messages” on page 1414

RTNMAP



Options



Notes:

[1] You can enter Options in any order between the parentheses.
[2] 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 and the default is NOHEADER.

Authorization

General User

Purpose

Use the RTNMAP command to display information about callable services library (CSL) routines currently loaded and bound to an address.

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 information is to be shown about all routines (this is the default) or the set of routines that satisfies the specified options.

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 information is displayed only for routines loaded with the USER option (on the RTNLOAD command). For more information, see Usage Note "1" on page 850.

SYstem
specifies information is displayed only for routines loaded with the SYSTEM option (on the RTNLOAD command). For more information, see Usage Note "1" on page 850.

GRoup *grpname*
specifies 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.

Header
specifies the output at the user's terminal is to be preceded by a header that identifies the columns of output displayed. For more information, see "Responses" on page 851. 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 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 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 this routine has been invoked by 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 by the CSLFPI macro, `count` is not incremented. Also, CMS commands that use CSLFPI 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.

The size of VMMTLIB members will always appear as zero unless they are loaded from a library generated by CSLGEN using the DASD option. To reduce minidisk space, the DASD version of

VMMTLIB is not supplied. Instead, the segment resident version is supplied in the VMMTLIB CSLSEG file. This file is loaded into user storage when generally the used segment is not found or the parameter MTSEG NO is used when IPLing. As is true with the segment resident form of VMMTLIB, no additional user storage can be recovered by removing a single routine. This is because the in-storage library must be loaded or removed as a whole.

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 the routine was loaded into using the RTNLOAD command.

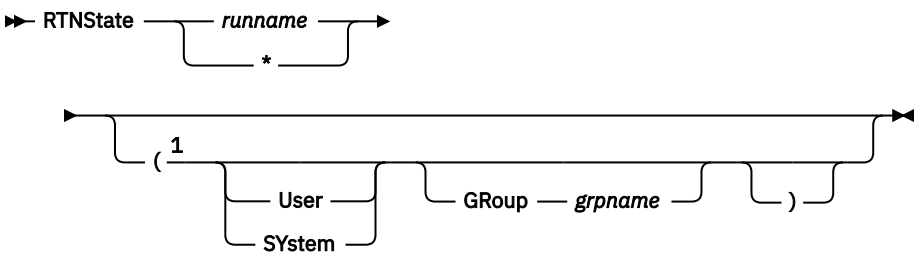
Messages and Return Codes

- DMS065E *option* option specified twice [RC=24]
- DMS066E *opt1* and *opt2* are conflicting options [RC=24]
- DMS1088E Routine *rtnname* cannot be mapped because it is not loaded [RC=28]
- DMS1088E Routine *rtnname* cannot be mapped because it was not loaded with the specified attribute [RC=28]
- DMS1088E Routine *rtnname* 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 their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

RTNSTATE



Notes:

[1] You can enter Options in any order between the parentheses.

Authorization

General User

Purpose

Use the RTNSTATE command to verify the existence of a loaded callable services library (CSL) routine.

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 a nonzero return code should be given, unless the named routine was loaded with the USER attribute. When used with an "*" for the name, this indicates a nonzero 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 a nonzero return code should be given, unless the named routine was loaded with the SYSTEM attribute. When used with an "*" for the name, this indicates a nonzero 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 a nonzero return code should be given, unless the named routine was loaded into the given group. When used with an "*" for the name, this indicates a nonzero 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. For more information, see "Examples" on page 854.
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 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 reloaded 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 reloaded 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 reloaded 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 reloaded into another group with the PUSH option, or if the routine was loaded into MYGROUP with the USER attribute and then reloaded 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 loaded with the USER attribute have been reloaded with the SYSTEM attribute and the PUSH option

4
if at least one routine was loaded with the USER attribute and then reloaded 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

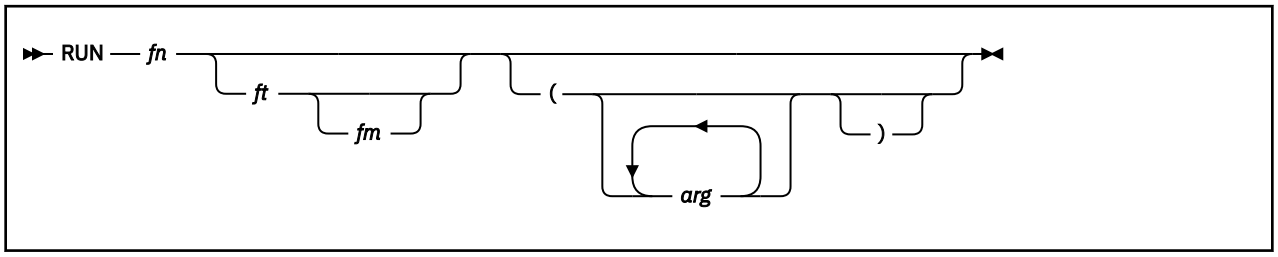
Messages and Return Codes

- DMS003E Invalid option: *option* [RC=24]
- DMS065E *option* option specified twice [RC=24]
- DMS066E *opt1* and *opt2* are conflicting options [RC=24]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

RUN



Authorization

General User

Purpose

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.

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 called is FORTVS2, 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 called is COBOL, TESTCOB, or COBOL2, whichever is found first. After successful compilation, the program text will be loaded and executed.

PLI or PLIOPT
The PLIOPT processor module is called to process the file for each of these file types. 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.

arg
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 REXX program, the arguments you enter on the RUN command line are available through 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 *z/VM: CMS Planning and Administration*.
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.

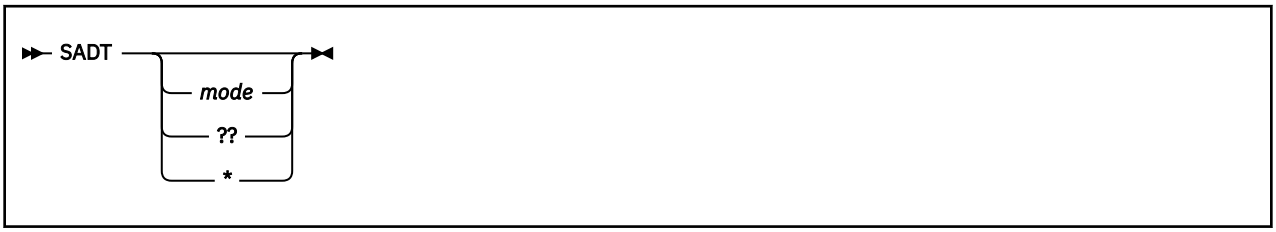
Messages and Return Codes

- DMS001E No filename specified [RC=24]
- DMS002E File *fn ft fm* not found [RC=28]
- DMS048E Invalid filemode *mode* [RC=24]
- DMS070E Invalid parameter *parameter* [RC=24]
- DMS999E No *ft* module found [RC=28]

An additional system message may be issued by this command. The reason for this message and its location is:

Reason	Location
Errors in the Shared File System	“File Pool Server Messages” on page 1414

SADT



Authorization

General User

Purpose

Use the SADT command to stack the pertinent information from the CMS ADT (active disk table) for the specified mode.

Operands

mode
is the access mode letter of the minidisk or SFS directory to be queried. There is no default for this operand.

??
stacks one line describing the R/W CMS minidisk or SFS directory with the most free space.

*****
stacks one line for each accessed CMS minidisk or SFS directory followed by a null line to indicate the end of the stacked data.

Usage Notes

1. The return code indicates the result of the query.
2. For accessed minidisks in CMS format and accessed SFS directories, a line is stacked that contains information about the minidisk or the directory.

   Note: A minidisk label may have imbedded blanks.

   The format of the stacked line is:

   Token
   Content

   1
   minidisk label or '-' for a directory

   2
   blocks used or '-' for a directory

   3
   blocks left or '-' for a directory

   4
   device address or 'DIR' for a directory

   5
   primary mode letter'

   6
   primary mode</<extension mode (if any)>>

7
number of files

8
number of cylinders or '-' for a directory or, if FBA dasd, the number of 512K blocks

9
device type (3380, and so on) or '-' for a directory

10
R/W or R/O (dependent on link and access)

3. A directory's free space is considered to be the same as its file pool's. If that file pool has the most free space and another of its directories is accessed, the directory accessed highest in the virtual machine's search order will be returned in response to the ?? operand.
4. You can also use the preferred QUERY DISK command to obtain similar information. See "QUERY DISK" on page 663 for details.
5. When calculating the blocks used (token 2) and blocks left (token 3) values for a minidisk, SADT does not factor in the minidisk blocks that are reserved for minidisk directory data and pointer blocks and for allocation map data and pointer blocks (the value of the ADTARES field in the minidisk's ADT). If this information is needed, use the preferred QUERY DISK command.

Messages and Return Codes

Return codes:

RC
Meaning

0
Command complete—no errors

1
Minidisk or directory is read-only (including read-only extensions)

2
Mode is not in use (nothing stacked)

4
OS disk (nothing stacked)

5
DOS disk (nothing stacked)

xx
where xx is RC from PARSECMD or CMSCALL

100
Explanation complete (when '?' specified)

System messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	"Command Syntax Error Messages" on page 1411

SAMGEN

Authorization

CMS Installer

Purpose

Use the SAMGEN command to load and save a saved segment (usually called CMSBAM) that contains the simulated VSE modules necessary to support Sequential Access Method (SAM) data management (DTFSD), the ESERV utility program, and Virtual Storage Extended/Virtual Storage Access Method (VSE/VSAM).

Usage Notes

1. Before the SAMGEN command can be used to load and save the saved segment, the CP DEFSEG command must be used to define the saved segment to CP.
2. To issue the SAMGEN command, you must be authorized to perform the CP SAVESEG operation.
3. SAMGEN performs the following operations:
   a. Fetches the simulated VSE phases from the CMSBAM DOSLIB file, which is supplied as part of VM
   b. Loads the simulated phases at the designated address
   c. Assigns a storage protection key of X'F'
   d. Saves the saved segment

Messages and Return Codes

- DMS363R ENTER LOCATION WHERE *sysname* WILL BE LOADED AND SAVED:
- DMS364I FETCHING *sysname* ...
- DMS365I SYSTEM *sysname* SAVED
- DMS366R ENTER NAME OF SYSTEM TO BE SAVED

SAMPNSS

Purpose

Use the SAMPNSS command to define the CMS named saved system.

Operands

cmsname
is the name to be defined for the CMS named saved system.

Usage Notes

The SAMPNSS command issues the CP DEFSYS command to define a skeleton system data file for CMS.

After using the SAMPNSS command to define the CMS named saved system, you need to IPL the CMS system minidisk with the SAVESYS *sysname* parameters to save the system:

```
ipl vdev clear parm savesys cmsname
```

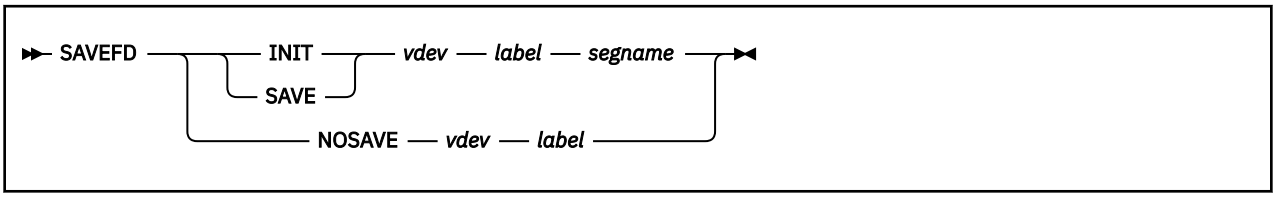
For example, to save CMS in a named saved system called CMS, issue the following command:

```
ipl 190 clear parm savesys cms
```

To save z/Architecture CMS (z/CMS) in a named saved system called ZCMS, issue the following command:

```
ipl 990 clear parm savesys zcms
```

SAVEFD



Authorization

Saved Segment Administrator

Purpose

Use the SAVEFD command to place file directory information for a CMS formatted minidisk into a saved segment.

SAVEFD places all file system control information needed to access the minidisk into the saved segment. This information includes File Status Table (FST) entries for each file on the minidisk, as well as other file system control data.

Note: This information includes a date and time stamp reflecting the last update to any file on the minidisk.

An alternative method for saving minidisk file directory information in shared storage is to use the SEGGEN command to save the information in a logical saved segment. For more information on segment spaces, DCSSs, and logical saved segments, see *z/VM: Saved Segments Planning and Administration*.

Operands

INIT
initializes the disk for a subsequent SAVEFD command. It accesses a disk and writes the saved segment name on the disk label record.

SAVE
saves the file directory information in a saved segment.

NOSAVE
disables the saved storage access. It accesses a disk and clears the saved segment name on the disk label record.

vdev
is the address of the disk for which saved file directory information is to be built (the disk must be linked R/W).

label
is a CMS minidisk label.

segname
is the name of the saved segment.

Saving File Directory Information in a DCSS

To put disk file directory information into shared storage on a z/VM system without using the packing support, follow these steps:

1. Define the saved segment for each disk whose file directory will be placed in shared storage using DEFSEG.

   Example

```
defseg fmn996 700-703 sr
defseg fmn997 800-803 sr
defseg fmn998 900-903 sr
defseg fmn999 a00-a03 sr
```

This creates a skeleton saved segment for each disk. Once this is done, you can use the SAVEFD command.

2. To write the saved segment name on the disk label record, enter:

```
savefd init vdev label segname
```

Example

```
savefd init 996 fmn996 fmn996
savefd init 997 fmn997 fmn997
savefd init 998 fmn998 fmn998
savefd init 999 fmn999 fmn999
```

In this example:

- The label is the same as the saved segment name, but it can be different.
- You can perform the SAVEFD INIT step before the DEFSEG step if you want.

3. To save the file directory information, enter:

```
savefd save vdev label segname
```

Example

```
savefd save 996 fmn996 fmn996
savefd save 997 fmn997 fmn997
savefd save 998 fmn998 fmn998
savefd save 999 fmn999 fmn999
```

Note: You need to do this step from a virtual machine with a virtual storage size greater than the residency definition of the saved segments you have defined. You cannot do a SAVEFD for a saved segment with page ranges not addressable by the virtual machine.

You have saved the directory content of the disks and have created new active saved segments. In this example, you used 4MB to contain the disk file directories. Once you link to disks with file directories that have been saved, you may enter:

```
access vdev filemode (saveonly
```

Example

```
access 996 d (saveonly
access 997 e (saveonly
```

Note:

1. Every time you update a SAVEFD minidisk, you must save its file directory by entering 'SAVEFD SAVE'.
2. You do not have to enter a DEFSEG or SAVESEG command again to save a copy of the disk file directory of a previously defined saved segment that has gone through a SAVEFD SAVE operation.
3. You do not have to enter 'SAVEFD INIT' if the saved segment name is unchanged.

Saving File Directory Information in a Member of a Segment Space

The following procedure should be used for initial installation of disk file directory information. For information on updating members of a segment space, see "1" on page 865.

To put disk file directory information into shared storage on a z/VM system using the packing support, follow these steps:

1. Define the saved segment for each disk with a file directory that will be placed in shared storage using DEFSEG.

   Example

   ```
   defseg fmn996 a00-a03 sr space fmnhelp
   defseg fmn997 a04-a07 sr space fmnhelp
   defseg fmn998 a08-a0b sr space fmnhelp
   defseg fmn999 a0c-a0f sr space fmnhelp
   ```

   This creates a skeleton saved segment for each disk and creates the segment space FMNHELP, which identifies the members of the space.
2. To write the saved segment name on the disk label record, enter:

   ```
   savefd init vdev label segname
   ```

   Example

   ```
   savefd init 996 fmn996 fmn996
   savefd init 997 fmn997 fmn997
   savefd init 998 fmn998 fmn998
   savefd init 999 fmn999 fmn999
   ```

   In this example:

   - The label is the same as the saved segment name, but it can be different.
   - You can perform the SAVEFD INIT step before the DEFSEG step if you want.
3. To save the file directory information in the saved segment, enter:

   ```
   savefd save vdev label segname
   ```

   Example

   ```
   savefd save 996 fmn996 fmn996
   savefd save 997 fmn997 fmn997
   savefd save 998 fmn998 fmn998
   savefd save 999 fmn999 fmn999
   ```

   Note: You need to do this step from a virtual machine with a virtual storage size greater than the residency definition of the saved segments you have defined. You cannot do a SAVEFD for a saved segment with page ranges not addressable by the virtual machine.

   You have saved the directory content of the disks and have created new active saved segments. In this example, you used 1MB to contain the disk file directories. Once you link to disks whose file directories have been saved, you may enter:

   ```
   access vdev filemode (saveonly
   ```

   Example

   ```
   access 996 d (saveonly
   access 997 e (saveonly
   ```

Note:

1. Each time you update a member of a segment space, you must redefine the entire segment space before entering the ‘SAVEFD SAVE’ command for the updated directory. You should use the SAME option to redefine the other members of the segment space. For example, if you want to place an updated copy in the saved segment for disks 996 and 998, enter the following:

   Example

   ```
   defseg fmn996 a00-a03 sr space fmnhelp
   defseg fmn997 same space fmnhelp
   defseg fmn998 a08-a0b sr space fmnhelp
   ```

```
defseg fmn999 same space fmnhelp
savefd save 996 fmn996 fmn996
savefd save 998 fmn998 fmn998
```

You have saved the directory content of the 996 and 998 disks and have created new active saved segments for FMN996, FMN998, and segment space FMNHELP. In this example, the update occupied the same page ranges (A00 through A03 and A08 through A0B), so you did not need to change the other members.

2. You do not have to enter 'SAVEFD INIT' if the saved segment name is unchanged.

For more information on using DEFSEG, see *z/VM: CP Commands and Utilities Reference*.

Usage Notes

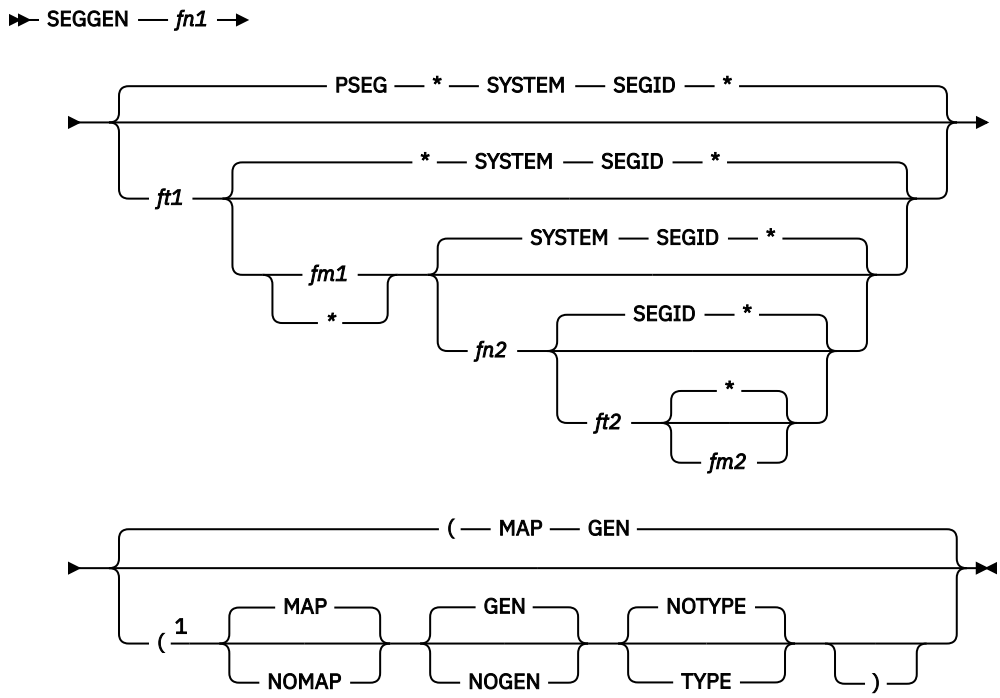
1. This command is no longer used during z/VM installation and service to build the HELP saved segment. The HELP saved segment is built as a logical saved segment in the HELPINST physical saved segment. For more information about building the HELP saved segment, see *z/VM: Installation Guide*.
2. Before the SAVEFD command can be used to load the saved segment, the CP DEFSEG command must be used to define the saved segment to CP.
3. To enter the SAVEFD SAVE command, you must be authorized to perform the CP SAVESEG operation.
4. To decide how big the saved segment should be for a SAVEFD minidisk, the following considerations should be made:
   - There is one 64-byte FST entry saved for each file on the minidisk (plus extra storage for the SAVEFD header and other file directory information).
   - The size of each segment is 1MB. Each 1MB segment may contain approximately 16,000 files of file directory information. So, if you were to define an entire segment for a minidisk file directory, you could save a file directory of a minidisk with up to approximately 16,000 files.
5. If you make a change to a SAVEFD minidisk, subsequent accessors of that minidisk will be unable to use the saved directory information until it has been resaved using SAVEFD SAVE. Therefore, you should avoid accessing a SAVEFD minidisk in read/write mode unless you are planning to subsequently resave the directory information.
6. The SAVEFD command accesses the SAVEFD minidisk as file mode Z. Therefore, any existing file mode Z is released when SAVEFD is entered.
7. Auxiliary directories are not supported on disks for a saved storage access. You should not use SAVEFD on disks that use auxiliary directories.
8. The saved segment must be defined below the 16MB line in order to save the file directory.
9. The CMS SEGGEN command may also be used to save file directory information in a segment.

Messages and Return Codes

- DMS014E Invalid function *function* [RC=24]
- DMS017E Invalid device address *vdev* [RC=24]
- DMS047E No function specified [RC=24]
- DMS050E Parameter missing after *value* [RC=24]
- DMS070E Invalid parameter *parameter* [RC=24]
- DMS109S Insufficient free storage available [RC=41]
- DMS126S Error {reading|writing} label on disk *mode(vdev)* [RC=100]
- DMS260E Disk not properly formatted for SAVEFD [RC=16]
- DMS283E The *name* saved segment could not be {found|saved}; return code *rc* from {SEGMENT|SAVESEG} [RC=128]
- DMS284E The saved segment is not completely inside the virtual machine [RC=40]
- DMS286E The saved segment is too small for the data being stored [RC=40]

- DMS288I *dcssname* saved segment not saved
- DMS653E Error executing *command*, rc=*nn* [RC=40]
- DMS1074S Disk not linked as R/W [RC=36]
- DMS1075E Label on disk *label* and label on command *label* do not match [RC=24]
- DMS1076E Segment name in disk label *segname* and segment name on command *segname* do not match [RC=24]
- DMS1077E Disk has not been initialized by SAVEFD INIT [RC=40]
- DMS1221E The *segname* saved segment must be below the 16MB line [RC=40]

SEGGEN



Notes:

[1] Options entered between the left and right parentheses can be in any order.

Authorization

Saved Segment Administrator

Purpose

Use the SEGGEN command to build and save a physical saved segment composed of one or more logical saved segments. SEGGEN uses the segment definitions in a physical segment definition file and one or more logical segment definition files, one for each logical saved segment to be included in the physical saved segment. For information about creating physical segment definition files and logical segment definition files, see *z/VM: Saved Segments Planning and Administration*.

The SEGGEN command updates or creates the system segment identification file. Each entry in this file associates a logical saved segment with its physical saved segment. The file must reside on the CMS system disk (usually file mode S) and must be named SYSTEM SEGID so it is available to CMS at initialization time. This allows CMS to recognize the logical saved segment name specified on the SEGMENT macro or SEGMENT command. After SEGGEN completes, you may have to copy the system segment identification file to the CMS system disk and resave CMS. For more information, see Usage Note “4” on page 870.

SEGGEN also generates two types of load map files, one for the physical saved segment and one for each logical saved segment within the physical saved segment.

Note: If the errors occur during SEGGEN processing, temporary files may be left on your read/write disk.

Operands

fn1
is the file name of the physical segment definition file. This name is also used as the name of the physical saved segment that is built.

ft1
is the file type of the physical segment definition file. The default file type is PSEG.

fm1
is the file mode of the physical segment definition file. The default file mode is *. If the file mode is not specified (or * is specified), all accessed file modes are searched.

fn2
is the file name of the system segment identification file. The default file name is SYSTEM.

ft2
is the file type of the system segment identification file. The default file type is SEGID.

fm2
is the file mode of the system segment identification file. The default file mode is *. For more information on the *fm2* operand and the destination of the new system segment identification file, see Usage Note "3" on page 869.

MAP
indicates segment load maps are to be produced. The default is MAP.

NOMAP
indicates no segment load maps are to be produced.

GEN
indicates the physical and logical saved segments are to be generated. The default is GEN. If an error occurs during SEGGEN processing, no saved segments are generated.

NOGEN
indicates the saved segments are not to be generated.

NOTYPE
indicates informational, warning, and error messages are suppressed by SEGGEN. The default is NOTYPE.

TYPE
indicates informational, warning, and error messages are not suppressed by SEGGEN.

Usage Notes

1. To use this command with the GEN option (which is the default), you must be authorized to perform the CP DEFSEG and SAVESEG operations.
2. The system segment identification file should be changed only by the SEGGEN command. Modification by any other means may cause unpredictable results.
3. If *fm2* is specified as a valid file mode, only a system segment identification file on that file mode will be used as a basis for creating the new file.

   If *fm2* is not specified (or * is specified), the first system segment identification file found in the disk search order is used as a basis for creating the new file.

   The destination of the new file is determined as follows:

   - If the file used as a basis was found on a read/write file mode, it is replaced by the new system segment identification file on the same file mode.
   - If the file used as a basis was found on a read/only file mode, the new system segment identification file is written on the first read/write file mode in the disk search order.
   - If no existing system segment identification file was found, a new one is created as follows:
     – If *fm2* is not specified or if * is specified, the new file is created on the first read/write file mode in the disk search order.

- If *fm2* is specified and it is a read/write file mode, the new file is created on that file mode. If *fm2* is a read/only file mode, error message DMS1178E is issued.

If there is no read/write file mode in the disk search order, error message DMS1178E is issued.

4. If you modify the contents of an existing logical saved segment (add, delete, or change data), you *do not* need to copy the system segment identification file to the CMS system disk.

   You *must* copy the system segment identification file to the CMS system disk only if you:

   - Create a new logical or physical saved segment
   - Delete an existing logical or physical saved segment
   - Change the relationship between the logical and physical saved segments (for example, if you move or copy a logical saved segment from one physical saved segment to another)

   You must also copy the system segment identification file to the test CMS system disk. This prevents the file from being regressed when you apply service. The z/VM service procedure always updates files on the test CMS system disk and then merges them to the production CMS system disk.

   Note: Only authorized user IDs can do the following functions. Contact your system support personnel.

   Access the production CMS system disk (usually MAINT 190) and the test CMS system disk (usually MAINT*vrm* 490) as alternate file modes, such as T and V:

   ```
   access 190 t
   access 490 v
   ```

   Then copy the system segment identification file to both disks. The file you place on the CMS system disks must be named SYSTEM SEGID, and the file mode number must be 2. Enter:

   ```
   copyfile fn2 ft2 fm2 system segid t2 (replace olddate
   copyfile fn2 ft2 fm2 system segid v2 (replace olddate
   ```

   Because copying a new or updated file to the CMS system disk changes the shared S-STAT (the saved S-disk file directory), you must also resave the CMS saved system. Enter:

   ```
   sampnss cms
   ipl 190 clear parm savesys cms
   ```

5. After using the SEGGEN command (and copying the system segment identification file to the CMS system disk, if required), you should erase or rename any file named SYSTEM SEGID on your accessed read/write disks so the latest version of SYSTEM SEGID is found only on the system disk.
6. If two logical saved segments of the same name are found in the system segment identification file, the one closest to the end of the file is the default logical saved segment used at initialization time. When a logical saved segment is associated with more than one physical saved segment, you can change the default association by using the SEGMENT ASSIGN command.
7. The SEGGEN command cannot be used to install an NLS segment above the 16MB line.
8. The physical saved segment built by SEGGEN must be defined in contiguous storage. For more information on defining saved segments in CP, see *z/VM: Saved Segments Planning and Administration*.
9. SEGGEN suppresses informational, error, and warning messages by default. If SEGGEN fails, rerun with the TYPE option specified to see all the messages.

Load Maps

Unless the NOMAP option is specified, the SEGGEN command creates or updates load maps on the first accessed read/write disk in the search order. The **physical segment load map** contains the starting address and name of each logical saved segment. The load map file is called *psegname* PSEGMAP *fm*, Where:

psegname
specifies the name of the physical saved segment

fm
specifies the read/write file mode used to create/update the system segment identification file

The **logical segment load map** contains the object name, the starting address of the object, the access mode, and other information about the directory or minidisk where the object was found. This load map file is called *lsegname* LSEGMAP *fm*, Where:

lsegname
specifies the name of the logical saved segment and

fm
specifies the read/write file mode used to create/update the system segment identification file.

Physical Segment Load Map

The format of the physical segment load map is:

```
address  lseg1
address  lseg2
   .       .
   .       .
   .       .
address  lsegn
SPACE UNUSED: nnnnn BYTES
```

address
specifies the starting address of the logical saved segment.

lseg1...lsegn
specifies the name of the logical saved segment.

SPACE UNUSED: nnnnn BYTES
specifies the amount of space remaining in the physical saved segment.

Logical Segment Load Map

The format of the logical segment load map is:

```
object   address   name  type   filemode vdev label
  .         .       .     .        .       - or -
  .         .       .     .        .     directory
  .         .       .     .        .       - or -
  .         .       .     .        .       txtlib
  .         .       .     .        .      .     .
  .         .       .     .        .      .     .
```

object
specifies the name of the object.

address
specifies the starting address of the object or the entry point addresses for a TEXT or MODULE.

name
specifies the entry point name of a TEXT or MODULE file or the file name specified for an exec.

type
specifies the type of object loaded, such as NUCEXT (nucleus extension) or SUBCOM (subcommand processor) for a TEXT or MODULE, EXEC or XEDIT for an exec, and so on.

filemode
specifies the file mode of the object when it was loaded.

vdev
specifies the virtual device address of the disk if the object was found on a disk.

label
specifies the label of the disk if the object was found on a disk.

directory
specifies the directory name if the object was found in a directory.

txtlib
specifies TXTLIB if the object was found in a TXTLIB.

Example of a Physical and Logical Load Map

Suppose you are building a physical saved segment named USERSEG that contains three logical saved segments called SEG1, SEG2, and SEG3. The physical segment load map, USERSEG PSEGMAP A5 is:

```
00101000 SEG1
00101500 SEG2
00102000 SEG3
SPACE UNUSED: FB000 BYTES
```

Three logical segment load maps are also produced. The load map for logical saved segment SEG1 would be named SEG1 LSEGMAP A5:

```
PROG1    00101000                      A 0199 DSK199
         00101000  PROG1E   NUCEXT
         00101050  PROG1S
PROG2    00101100                      A 0199 DSK199
         00101100  PROG2    SUBCOM
EXEC1    00101200                      D 0192 EXE192
                   EXEC1    EXEC
EXEC2    00101300                      D 0192 EXE192
                   EXEC2    EXEC
PROGLIB  00101400                      B PROG.MATH.LIB
```

A physical saved segment can have the same name as one of its logical saved segments without a file naming conflict occurring. However, you or your system administrator should avoid this situation because the SEGMENT LOAD command and macro look for a logical saved segment first. If a physical saved segment and logical saved segment have the same name, the logical saved segment is loaded and the physical saved segment would never be loaded.

Messages and Return Codes

- DMS002E File *fn ft fm* not found [RC=28]
- DMS073E Unable to open file *fn ft fm* [RC=28]
- DMS104S Error *nn* reading file *fn ft fm* [RC=100]
- DMS105S Error *nn* writing file *fn ft fm* [RC=100]
- DMS283E The *name* saved segment could not be reserved; return code *rc* from SEGMENT RESERVE [RC=128]
- DMS283E The *name* saved segment could not be saved; return code *rc* from SAVESEG [RC=128]
- DMS283E The *name* saved segment could not be found; return code *rc* from SEGMENT [RC=128]
- DMS284E The saved segment is not completely inside the virtual machine [RC=88]
- DMS639E Error in *routine* routine; return code was *nnnn* [RC=256]
- DMS1178E No read/write disk or directory with space is accessed [RC=36]
- DMS1279E Error(s) occurred during SEGGEN processing. [RC=32]
- DMS1280E Segment *name* is already defined as a logical segment. [RC=40]
- DMS1281E Errors writing to System Segment Identification file. Segment was not saved. [RC=100]
- DMS1282E Segment cannot span 16MB boundary. [RC=40]
- DMS1283E Unexpected end of file encountered [RC=32]

Additional system messages may be issued by this command.

These messages may be written to the physical segment map file:

- File *fn ft fm* not found
- Unable to open file *fn ft fm*
- A not valid record of: *record*
- Unexpected end of file encountered
- Logical segment *segname* already exists in physical segment.
- SPACE UNUSED: *length* BYTES

- Error encountered in logical segment profile *fn ft fm*. Return code *rc*
- Error encountered in logical segment epifile *fn ft fm*. Return code *rc*
- Error(s) occurred while processing logical segment definition file *fn ft fm*

The following messages may be written to the logical segment map file:

- Unable to open file *fn ft fm*
- Unexpected end of file encountered
- Not enough room for logical segment index entry in working storage
- Insufficient free storage available to build logical segment index
- Not enough room for logical segment index in saved segment

For MODULE records:

- Module (*fn ft*) not found
- Module (*fn ft*) must be relocatable
- Not enough room for module (*fn ft*) in saved segment
- Module or text (*fn ft*) already exists in logical segment
- Module (*fn ft*) load error: return code=*rc*

For TEXT records:

- TEXT (*fn ft*) not found
- Not enough room for text (*fn ft*) in saved segment
- Missing END statement
- Missing BEGIN statement
- TEXT (*fn ft*) load error: return code=*rc*
- Unable to resolve specified entry point
- Conflicting BEGIN/END encountered
- Module or text (*fn ft*) already exists in logical segment
- A not valid record within a BEGIN/END block

For EXEC records:

- EXEC (*fn ft*) load error: return code=*rc*
- Not enough room for EXEC (*fn ft*) in saved segment
- EXEC-2 EXEC (*fn ft*) cannot be loaded above 16MB
- EXEC (*fn ft*) already exists in logical segment

For LIBRARY records:

- Library (*fn ft*) not found
- Library (*fn ft*) not relocatable
- Not enough room for library (*fn ft*) in saved segment
- Library (*fn ft*) already exists in logical segment
- Library (*fn ft*) load error: return code=*rc*

For LANGUAGE records:

- Language file (*fn ft*) not found
- Not enough room for language file (*fn ft*) in saved segment
- Language information (*fn ft*) already exists in logical segment
- Language file (*fn ft*) cannot be loaded
- Language (*fn ft*) load error: return code=*rc*

For DISK records:

- DISK must be the only object in a logical segment
- Disk information load error: return code=*rc*
- Skip records cannot be placed before a DISK record
- Saved segment containing disk information must be below 16Mb
- Unable to access disk
- Disk must be linked R/W to perform INIT
- Disk must be EDF format
- Label on disk and label in LSEG file do not match
- Error writing label on disk
- Not enough room for disk information in saved segment

For USER records:

- User object (*fn ft*) load error: return code=*rc*
- Not enough room in segment for user object (*fn ft*)

For SKIP records:

- Not enough room in segment to perform skip

SEGMENT

Authorization

General User

Purpose

Use the SEGMENT command to manage saved segments in your virtual machine. The SEGMENT command allows you to:

- Reserve CMS storage for a saved segment
- Assign a logical saved segment to a physical saved segment
- Load a saved segment into storage
- Purge a saved segment
- Release storage previously reserved for a saved segment

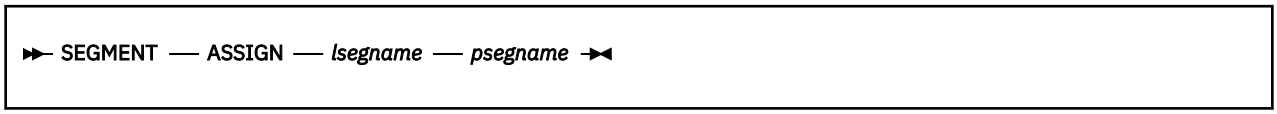
Note: You cannot use the SEGMENT command to manage discontiguous saved segments (DCSSs) that include page addresses above 2047 MB. You must use DIAGNOSE code X'64'. For more information, see *z/VM: CP Programming Services*.

These are the five functions of the SEGMENT command:

Command	Location
SEGMENT ASSIGN	“SEGMENT ASSIGN” on page 876
SEGMENT LOAD	“SEGMENT LOAD” on page 877
SEGMENT PURGE	“SEGMENT PURGE” on page 881
SEGMENT RELEASE	“SEGMENT RELEASE” on page 883
SEGMENT RESERVE	“SEGMENT RESERVE” on page 885

There is also a SEGMENT macro. For reference information on the SEGMENT macro, see *z/VM: CMS Macros and Functions Reference*. For usage information on the SEGMENT command, see *z/VM: CMS Application Development Guide*. For usage information on the SEGMENT macro, see *z/VM: CMS Application Development Guide for Assembler*.

SEGMENT ASSIGN



Purpose

Use the SEGMENT ASSIGN command to change the physical saved segment from which a logical saved segment will be used.

Operands

lsegname
is the 1-8 character name of the logical saved segment.

psegname
is the 1-8 character name of the physical saved segment.

Examples

You have two logical saved segments with the same name, EXECS, residing in two physical saved segments, PSEG1 and PSEG2. If you want to use the EXECS saved 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 saved segment in PSEG2 is used.

Messages and Return Codes

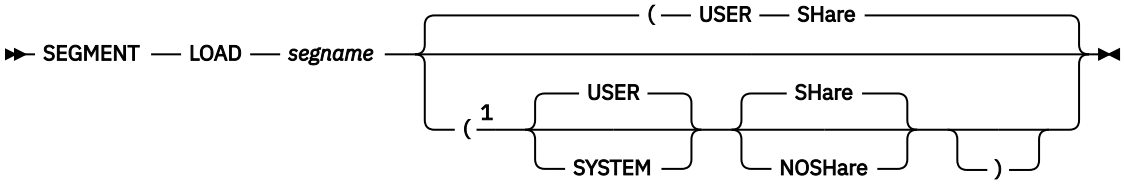
- DMS109S Virtual storage capacity exceeded, return code *rc* from storage management [RC=104]
- DMS283E The *segname* could not be loaded|reserved|released|purged; return code *rc* from SEGMENT. [RC=*rc*]
- DMS1274E Logical segment *lsegname* does not exist in physical segment *psegname*. [RC=28]
- DMS1275E Logical segment *segname* is currently active and cannot be assigned. [RC=36]
- DMS1277E Logical segment *segname* does not exist. [RC=28]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

Note: For message DMS283E, the return code *rc* is that of DIAGNOSE X'64'. For more information, see *z/VM: CP Programming Services*.

SEGMENT LOAD



Notes:

[1] You can enter Options in any order between the parentheses.

Purpose

Use the SEGMENT LOAD command to load a saved segment into your virtual storage.

Operands

segname
is the 1-8 character name of the logical saved segment, physical saved segment, or CP segment space to be loaded.

Options

USER
specifies the saved segment is to be released during ABEND processing. This is the default.

SYSTEM
specifies the saved segment is not to be released during ABEND processing. If you do not specify SYSTEM and ABEND recovery is invoked, the segment storage space is released.

If you already reserved storage space for the saved segment with the SYSTEM option on the SEGMENT RESERVE command, that option overrides the value specified on the SEGMENT LOAD command.

For more information, see Usage Note “8” on page 878.

SHare
loads a shared copy of the saved segment. This is the default.

NOSHare
loads a nonshared copy of the saved segment.

For more information, see Usage Note “9” on page 878.

Usage Notes

1. The SEGMENT LOAD command is an alternative to the DIAGNOSE code X'64' LOADSYS function. However, you cannot use the two interchangeably.
2. You can load saved segments even if they reside within the virtual machine's address space. However, the CMS storage must be free storage; that is, no programs or data may already reside there.
3. 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 command. If one is found, storage for the associated physical saved segment is reserved (if not already reserved). If the physical saved segment is a member of a CP segment

space, storage is reserved for the entire segment space. Then the storage space is loaded (if not already loaded), and the contents of the logical saved segment are processed.

b. If a logical saved segment with that name is not found, CMS searches the list of storage spaces previously reserved with the SEGMENT RESERVE command to determine if a space has been reserved for a saved segment with the requested name. If one is found, the storage space is loaded (if not already loaded).

c. If no reserved storage space exists, CMS determines whether the requested saved segment has been defined in CP. If the saved segment has been defined in CP, CMS issues a SEGMENT RESERVE command to create a reserved storage space, then loads the saved segment. If the saved segment is a member of a CP segment space, CMS reserves storage space for and loads the entire segment space.

d. If the requested saved segment is none of the above, the command returns a return code of 44.

This process allows an application to be loaded even if the saved segment resides within the virtual machine.

4. When a logical saved 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 information is processed
   - User exit routines are called

   Objects in other logical saved segments in the physical saved segment are not processed.
5. The language of application message repositories must match the current system language to be added to the active set of applications.
6. Nucleus extensions, subcommand processors, and execs established by SEGMENT LOAD override objects of the same name. Nucleus extensions in saved 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.
7. The SHARE attribute of a physical saved segment that contains logical saved segments is set by the first logical saved segment that is loaded. If other logical saved segments in that physical saved segment are loaded, they cannot change the SHARE attribute. All logical saved segments in the same physical saved segment have the same SHARE attribute. If you specify a SHARE or NOSHARE option that does not match the SHARE attribute of the physical saved segment, the saved segment is not loaded and a nonzero return code is returned.
8. If the saved segment includes a callable services library (CSL), individual CSL routines loaded with the SYSTEM option on the RTNLOAD command are not protected from abend processing unless the saved segment is loaded with the SYSTEM option.
9. If the nonshared copy of the saved segment is to be loaded outside the maximum storage size of the virtual machine (as defined on the USER or IDENTITY directory statement), the saved segment must be identified on a NAMESAVE control statement in the user's directory entry. If *segname* is a physical saved segment that is a DCSS, or a logical saved segment contained in a physical saved segment that is a DCSS, the name identified on the NAMESAVE statement is the DCSS. If *segname* is a physical saved segment that is a member of a CP segment space, or a logical saved segment contained in a physical saved segment that is a member of a segment space, the name identified on the NAMESAVE statement must be the segment space.
10. Loading a physical saved segment does not give you access to the logical saved segments it contains. Loading a CP segment space does not give you access to its members. Use the SEGMENT LOAD command to load the specific saved segments you need.
11. In rare cases, a SEGMENT LOAD request could be delayed. This could happen if another user is using the CP SPXTAPE DUMP command to dump the same saved segment name onto tape. The delay will occur only if the system data file containing the saved segment was not loadable when the dump

began, or if all other system data files with the same name become not loadable during the dump. The LOAD results will depend on whether the system data file is loadable when the delay ends.

A system data file is not loadable if any of the following conditions are true:

- It is a skeleton.
- It is class P (pending purge).
- It is a member of a segment space whose system data file is a skeleton because at least one member of the space is a skeleton.
- It is a member of a segment space that is missing one or more members, resulting from either of the following:
  - The CP PURGE NSS command was used without the ASSOCIATES operand to purge the members.
  - Only some of the system data files were loaded from tape by the CP SPXTAPE LOAD command.

Messages and Return Codes

- DMS109S Virtual storage capacity exceeded, return code *rc* from storage management [RC=104]
- DMS283E The *segname* could not be loaded|reserved|released|purged; return code *rc* from SEGMENT. [RC=*rc*]
- DMS338E Saved segment *segname* could not be reserved or loaded because the skeleton segment is already reserved [RC=36]
- DMS343E Storage in range *address-address* for *name* in use. [RC=41]
- DMS1083E Saved segment *segname* does not exist [RC=44]
- DMS1266E Error occurred while loading logical segment *segname*, return code *rc* [RC=256]
- DMS1267E Error occurred while loading user objects, return code *rc* [RC=256]
- DMS1270E The SHARE/NOSHARE option specified does not match the SHARE attribute of the containing physical segment. [RC=36]
- DMS1271E *name* contains reserved and/or loaded logical segments and cannot be reserved, loaded, or purged. [RC=36]
- DMS1272E Physical segment *segname* is already active. [RC=36]
- DMS1295E *name* segment space contains reserved or loaded member saved segments and cannot be reserved or loaded. [RC=36]
- DMS1296E *name* member saved segment cannot be reserved or loaded in a segment space that is already reserved or loaded. [RC=36]
- DMS3992E Not authorized to load a nonshared copy of *segname* [RC=*rc*]
- DMS3993E *segname* saved segment can not be loaded beyond 16M. [RC=*rc*]
- DMS3994E *segname* member saved segment mode differs from the segment space mode. [RC=*rc*]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

Note: For messages with return code shown as *rc*, the return code is that of DIAGNOSE X'64'. For more information, see *z/VM: CP Programming Services*.

Return codes:

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 code X'64'.

SEGMENT PURGE

Purpose

Use the SEGMENT PURGE command to remove a saved segment from your virtual storage.

Operands

segname
is the 1-8 character name of the logical saved segment, physical saved segment, or CP segment space to be purged.

Usage Notes

1. If the SEGMENT LOAD command reserved storage space for a saved segment, the SEGMENT PURGE command automatically releases that storage space.
2. If the saved segment being purged is a logical saved segment, all the objects in the saved segment are purged:
   - User exit routines are called
   - Nucleus extensions and execs are dropped
   - Subcommand processors are cleared
   - Language information is deleted
   - Libraries are removed from the list of callable services libraries.
   - Minidisks with access information in the segment are released

   If the purged logical saved segment is the only active saved segment in the segment storage space, the storage space is released.
3. Use the SEGMENT PURGE command to purge a saved segment that was loaded using the SEGMENT LOAD command. If a saved segment was loaded using the DIAGNOSE code X'64' LOADSYS function, do not use SEGMENT PURGE to purge this saved segment. Use the DIAGNOSE code X'64' PURGESYS function.

   You cannot use the SEGMENT PURGE command and the DIAGNOSE code X'64' PURGESYS function interchangeably.

Messages and Return Codes

- DMS109S Virtual storage capacity exceeded, return code *rc* from storage management [RC=104]
- DMS283E The *segname* could not be loaded|reserved|released|purged; return code *rc* from SEGMENT. [RC=*rc*]
- DMS345E *segname* was not loaded via SEGMENT LOAD function [RC=40]
- DMS1083E Saved segment *segname* does not exist [RC=44]
- DMS1268E Error occurred while purging logical segment *segname*, return code *rc* [RC=256]
- DMS1269E Error occurred while purging user object *name*, return code *rc* [RC=256]
- DMS1271E *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 their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

Note: For message DMS283E, the return code *rc* is that of DIAGNOSE X'64'. For more information, see *z/VM: CP Programming Services*.

Return codes:

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 code X'64'.

SEGMENT RELEASE

Purpose

Use the SEGMENT RELEASE command to return storage to CMS that was previously reserved for a saved segment.

Operands

segname
is the 1-8 character name of the logical saved segment, physical saved segment, or CP segment space for which the reserved storage is to be released.

Usage Notes

If *segname* is a:

- Logical saved segment, it is removed from the list of reserved logical saved segments. If the physical saved segment that contains *segname* no longer has any logical saved segments loaded or reserved, the physical saved segment is removed from the virtual machine and the reserved storage is returned to CMS. If *segname* is a loaded logical saved segment containing minidisk directory information, the minidisk will also be released.
- Physical saved segment, all the loaded or reserved logical saved segments within *segname* are released first, and then *segname* is released. If *segname* is a member of a CP segment space, and the segment space no longer has any member saved segments loaded or reserved, the segment space is released and the storage is returned to CMS.
- CP segment space, and if any members of *segname* are physical saved segments that contain logical saved segments, all the loaded or reserved logical saved segments are released. Then the members of *segname* are released. Then *segname* is released and the storage is returned to CMS.

You can reserve or load several logical saved segments within a physical saved segment, and then reclaim all the storage used by these saved segments by issuing the SEGMENT RELEASE command for the physical saved segment.

Messages and Return Codes

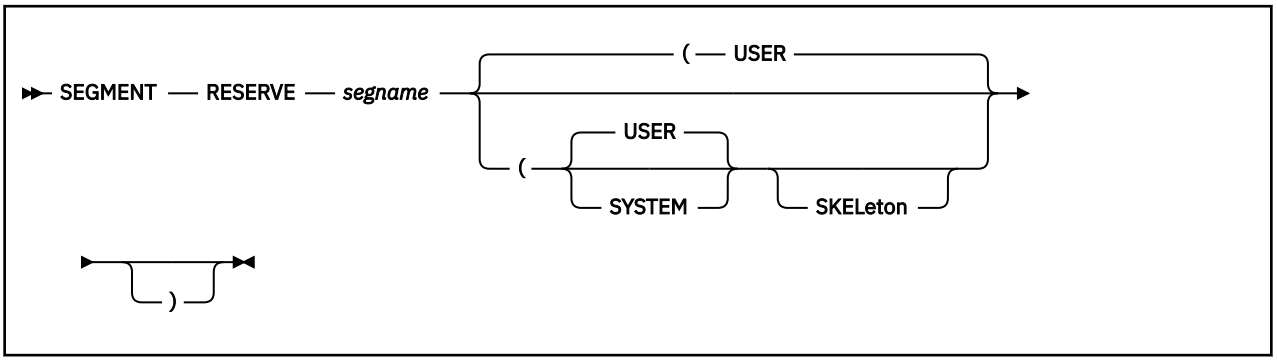
- DMS109S Virtual storage capacity exceeded, return code *rc* from storage management [RC=104]
- DMS283E The *segname* could not be loaded|reserved|released|purged; return code *rc* from SEGMENT. [RC=*rc*]
- DMS344E Segment space *name* has not been reserved [RC=40]
- DMS345E *segname* was not loaded via SEGMENT LOAD function [RC=40]
- DMS1083E Saved segment *segname* does not exist [RC=44]
- DMS1268E Error occurred while purging logical segment *segname*, return code *rc* [RC=256]
- DMS1269E Error occurred while purging user object *name*, return code *rc* [RC=256]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

Note: For message DMS283E, the return code *rc* is that of DIAGNOSE X'64'. For more information, see *z/VM: CP Programming Services*.

SEGMENT RESERVE



Purpose

Use the SEGMENT RESERVE command to reserve a CMS storage space for a saved segment so a subsequent SEGMENT LOAD command or macro, or a DIAGNOSE code X'64', may be safely entered. SEGMENT RESERVE does not actually load the saved segment into storage.

Use the SEGMENT RESERVE command with the SKELETON option to reserve a CMS storage space for a skeleton segment (the class S NSS) so a segment building utility may be used effectively.

Operands

segname
is the 1-character to 8-character name of the logical saved segment, physical saved segment, or CP segment space for which storage is to be reserved.

Options

SYSTEM
specifies the storage space is not to be released during ABEND processing. If this option is not specified, the storage space is released if ABEND recovery is invoked.

USER
specifies the storage space is to be released during ABEND processing. This is the default.

SKELeton
specifies storage is to be reserved for the skeleton segment.

Usage Notes

1. The location and size of the reserved storage space is determined by the location and size of the corresponding saved segment defined in CP.
   - If *segname* is a physical saved segment that is a DCSS, or if *segname* is a logical saved segment contained in a physical saved segment that is a DCSS, storage is reserved for the DCSS.
   - If *segname* is a physical saved segment that is a member of a CP segment space, or if *segname* is a logical saved segment contained in a physical saved segment that is a member of a segment space, storage is reserved for the entire segment space.
   - If the SKELETON option was specified, storage is reserved only for the skeleton segment (class S NSS) that corresponds to *segname*, even if *segname* is a member of a segment space.
2. In rare cases, a SEGMENT RESERVE request could be delayed. This could happen if another user is using the CP SPXTAPE DUMP command to dump the same saved segment name onto tape. The delay will occur only if the system data file containing the saved segment was not loadable when the dump

began, or if all other system data files with the same name become not loadable during the dump. The RESERVE results will depend on whether the system data file is loadable when the delay ends.

A system data file is not loadable if any of these conditions are true. It is:

- A skeleton.
- Class P (pending purge).
- A member of a segment space with a system data file that is a skeleton because at least one member of the space is a skeleton.
- A member of a segment space that is missing one or more members, resulting from either of the following:
  - The CP PURGE NSS command was used without the ASSOCIATES operand to purge the members.
  - Only some of the system data files were loaded from tape by the CP SPXTAPE LOAD command.

3. When the SKELETON option is specified, the SEGMENT command will not search for logical saved segments. Only physical segments can be reserved with the SKELETON option.

Messages and Return Codes

- DMS109S Virtual storage capacity exceeded, return code *rc* from storage management [RC=104]
- DMS283E The *segname* could not be loaded|reserved|released|purged; return code *rc* from SEGMENT. [RC=*rc*]
- DMS338E {Saved|Skeleton} segment *segname* could not be reserved [or loaded] because the {saved|skeleton} segment is already reserved [or loaded]. [RC=36]
- DMS358E {Saved|Skeleton} segment *name* has already been reserved [RC=4]
- DMS1083E Saved segment *segname* does not exist. [RC=44]
- DMS1270E The SHARE | NOSHARE option specified does not match the SHARE attribute of the containing physical segment [RC=24]
- DMS1271E *name* contains reserved and/or loaded logical segments and cannot be reserved, loaded, or purged. [RC=36]
- DMS1272E Physical segment *segname* is already active. [RC=36]
- DMS1295E *name* segment space contains reserved or loaded member saved segments and cannot be reserved or loaded. [RC=36]
- DMS1296E *name* member saved segment cannot be reserved or loaded in a segment space that is already reserved or loaded. [RC=36]
- DMS343E Storage in range *address-address* for *name* in use. [RC=41]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

Return codes:

RC
Meaning

0
Normal.

1
The saved segment is defined as a VMGROUP and cannot be loaded with SEGMENT LOAD.

24
Invalid parameter was specified.

40
Segment is not loaded.

44
Segment does not exist.

174
A paging I/O error occurred during the FIND operation.

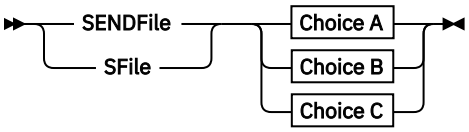
256
Error processing contents of logical segment.

304
CP has detected an unacceptable condition. This condition is most likely to occur in the event of a CP soft ABEND.

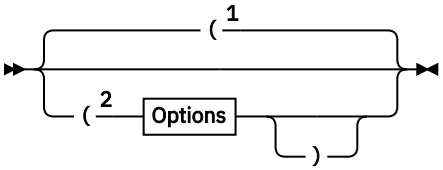
Other return codes may be issued by DIAGNOSE code X'64'.

Note: For message DMS283E, the return code *rc* is that of DIAGNOSE X'64'. For more information, see *z/VM: CP Programming Services*.

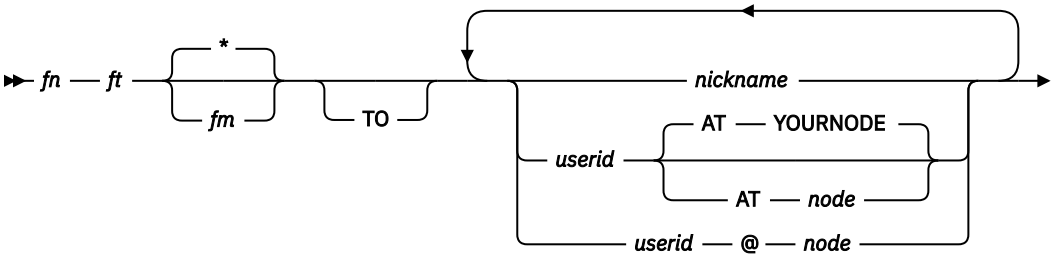
SENDFILE

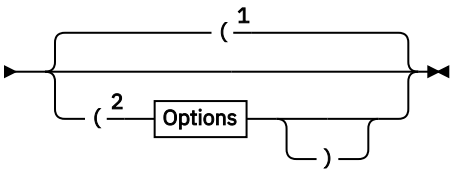


Choice A

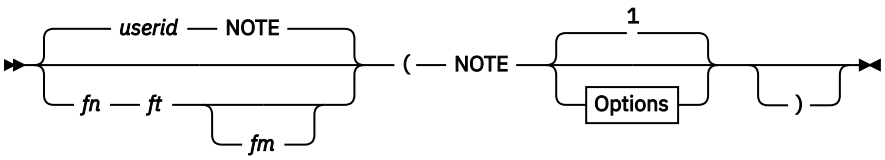


Choice B





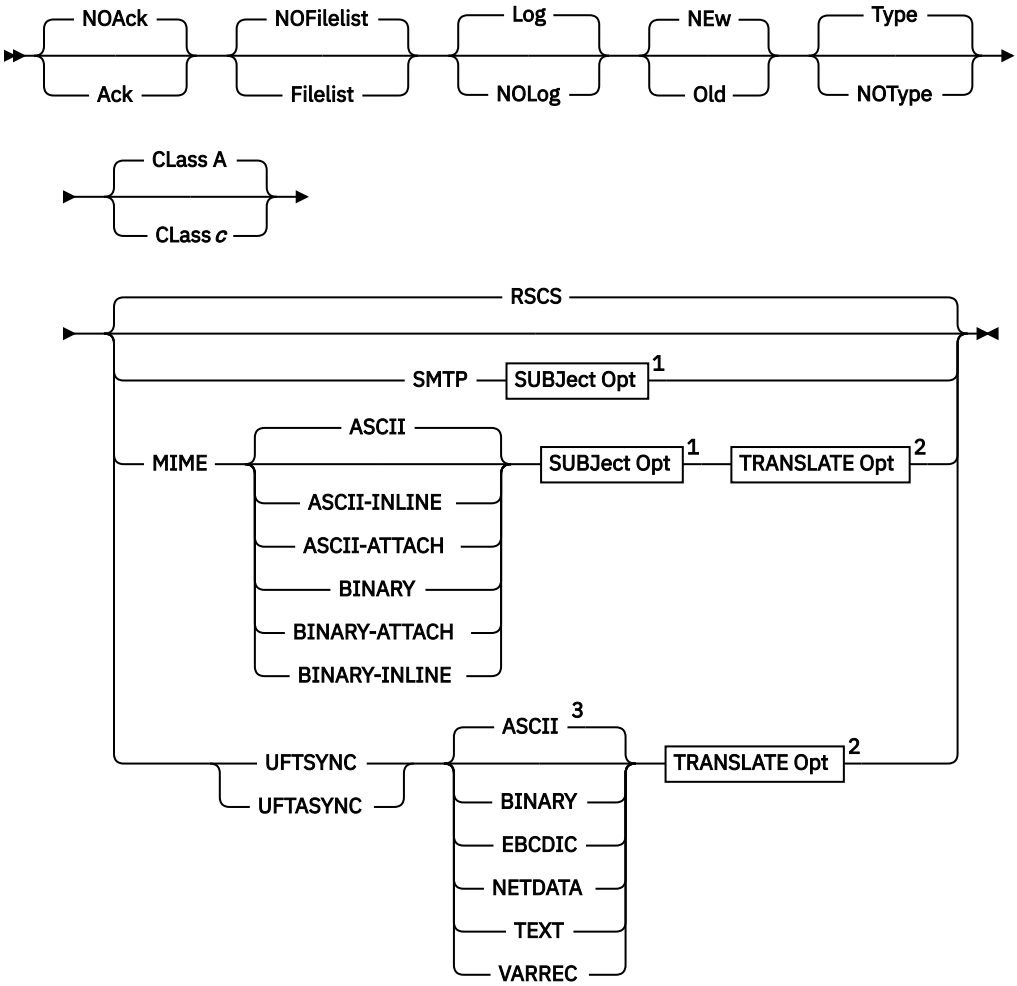
Choice C



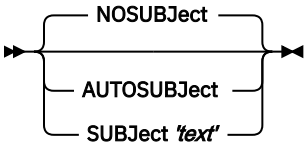
Notes:

[1] The defaults you receive appear above the line in the Options fragment.
[2] You can enter Options in any order between the parentheses.

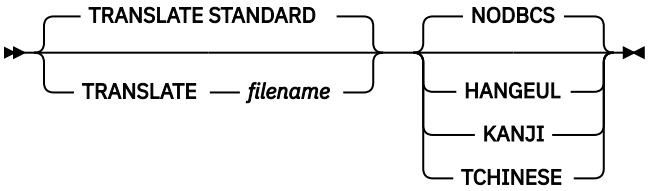
Options



SUBJect Options



TRANSLATE Options



Notes:

[1] If this option is specified with anything other than SMTP or MIME, it is ignored. It is also ignored in all cases if using the NOTE option.

[2] These are valid only when UFTSYNC ASCII, UFTSYNC TEXT, UFTASYNC ASCII, UFTASYNC TEXT, MIME ASCII, or MIME ASCII-INLINE has been specified.

[3] If conflicting options are specified, the last option specified is used.

Authorization

General User

Purpose

Use the SENDFILE command to send files or notes to one or more computer users on your computer or on other computers.

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 “NAMES” on page 535.

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 T0 is used.

TO
is a keyword operand. It can be omitted if the first "name" is not T0.

nickname
userid
userid* AT *node
specifies one or more recipients to whom the file is to be sent. If the same recipient is specified more than once, the user receives only one copy of the file.

- A *nickname* is a short form of a user ID you have defined in your *userid* NAMES file, where *userid* identifies your user ID. This *nickname* may represent a single person (on your computer or another computer), or a list of several people. If a *nickname* cannot be found in the *userid* NAMES file, it is assumed to be a fully-specified user ID of someone on your computer. For more information on nicknames, see “NAMES” on page 535.
- A *userid* identifies a user on your system or another system (*node*).

  A user ID cannot be "AT" or “CC:”.
- A *node* identifies any RSCS node name or TCP/IP host name or IP address. The rule used for determining a node be treated as a TCP/IP address is that it contains a colon, a period or a dot. You may use the TCPEXIT EXEC to change these rules to fit your environment. If the node is determined to be a TCP/IP address it will be treated as such so long as an option other than RSCS is specified. A TCP/IP host name that ends with a period will have the local domain name appended to it in order to construct a fully-qualified host name.

  Non-TCP/IP destinations are sent through CMS or RSCS as necessary, even on the same SENDFILE command, as long as an option other than RSCS is specified. You may freely intermix the *userid* and *nickname* forms to specify recipients.

  You may freely intermix all forms to specify recipients. The UFTASYNC, UPTSYNC, and MIME options control how the file is or files are sent to TCP/IP Internet hosts. The options are ignored for RSCS or local addresses.

  If no operands are specified with SENDFILE, a menu is displayed. For more information, see Usage Note “2” on page 893, "Using the SENDFILE MENU".

NOTE
specifies the file is to be sent as a note, this file was created by using the NOTE command. The T0 operand and the list of user IDs or nicknames 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.)

You may freely intermix the *userid* and *nickname* forms to specify recipients. If no operands are specified with SENDFILE, a menu is displayed. For more information on this menu, see Usage Note “2” on page 893, "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. This option is ignored for TCP/IP destinations.

NOAck
requests no acknowledgment be returned when the recipient RECEIVEs a file. This is the default.

Filelist
specifies 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 sends 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 “FILELIST” on page 291, Usage Note “7” on page 298, "Saving a List of Files".

NOFilelist
specifies the file is not a list of files. This is the default.

Log
specifies 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. This is the default.

NOLog
specifies this file transmission is not to be logged.

NEw
specifies 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 or NETDATA to read the file. This option is ignored for TCP/IP destinations. This is the default.

Old
specifies the file is sent using DISK DUMP. This option should be specified when the recipient of the file does not have the RECEIVE or NETDATA commands available to read the file. When OLD is specified, no acknowledgment (the ACK option) can be sent. This option is ignored for TCP/IP destinations.

Type
specifies the files sent and the user IDs and nodes to which the files were sent are displayed at the terminal. This is the default.

NOType
specifies no information is to be displayed.

CLass *c*
specifies the spool class to be used when sending the file. The operand *c* is a 1-character alphanumeric field whose value can be A through Z, 0 through 9, or equal sign (=). If an equal sign is specified, the current spool class of the punch is used.

RSCS
Network addresses should only be processed for RSCS networks, no TCP/IP addresses are accepted, and files will not be sent through TCP/IP services. This is the default.

SMTP
specifies the file being sent is using SMTP without any encoding or MIME headers for TCP/IP addresses. For RSCS addresses, the file is sent using RSCS.

MIME
specifies the file being sent is using SMTP for TCP/IP addresses. The file is encoded in BASE64 and sent as a single-part MIME data stream. For RSCS addresses, the file is sent using RSCS.

NOSUBJect
specifies the file being sent will have a blank subject line associated with it. This is the default.

AUTOSUBJect
specifies SENDFILE should set the subject line to be associated with the file being sent. The subject to be set is determined the following way:

- The contents of the file being sent is examined. If the first line of the file is Subject:; Subj: or Re:, this is used to set the subject.
- Otherwise, it sets the subject to "File:" *fn ft* "from User ID" *userid*.

SUBJect *'text'*
specifies the file being sent will use the 'text' passed in as the subject line to be associated with it. The subject text must be entered in single quotation marks.

UFTSYNC
sends the file to the remote host's UFT (Unsolicited File Transfer) server directly from the user's virtual machine for TCP/IP addresses. If no UFT server is available on the remote host, the file will be sent using SMTP instead. The file is sent as a two-part MIME data stream. The first MIME part is a metafile which contains all the UFT information associated with the file (file id, format, record length, and so forth). The second MIME part is the file data encoded in Base64 (a MIME-defined encoding format). TCP/IP addresses must be IPv4 addresses.

UFTASYNC
sends the file to the server, identified in TCPIP DATA, where it will be sent to the remote host's UFT server for TCP/IP addresses. For RSCS addresses, the file is sent using RSCS. See Usage Note “11” on page 898 for additional details. TCP/IP addresses must be IPv4 addresses.

ASCII
TEXT
ASCII-INLINE
specifies the user's file is to be translated from EBCDIC to ASCII format as it is being transferred. This option should be used for files that are network plain text, generally anything human readable. This is the default. This option is used only if MIME, UFTASYNC, or UFTSYNC is specified. For the MIME option, the file is sent inline.

ASCII-ATTACH
specifies the user's file is to be translated from EBCDIC to ASCII format as it is being transferred. This option should be used for files that are network plain text, generally anything human readable. This option is used only if MIME is specified. The file is sent as an attachment.

BINARY
BINARY-ATTACH
specifies the file is to be transferred as binary data (no translation). Specifies the user's file is to be transferred as an octet stream of bytes for which record boundaries are not important. For example, the GIF or TAR files. This option is used only if MIME, UFTASYNC or UFTSYNC is specified. For the MIME option, the file is sent as an attachment.

BINARY-INLINE
specifies the file is to be transferred as binary data (no translation). Specifies the user's file is to be transferred as an octet stream of bytes for which record boundaries are not important. For example, the GIF or TAR files. This option is used only if MIME is specified. The file is sent inline.

EBCDIC
specifies the file is to be transferred in EBCDIC format.

NETDATA
specifies the user's file is to be transferred in a NETDATA format. The file will arrive at the destination in NETDATA format, so this format is only useful when sending files to systems that have NETDATA capabilities.

VARREC
specifies the user's file is to be transferred and record boundaries are important, such as MODULE files. The file is transferred with a two byte length preceding each record to allow the receiving system to keep the record boundaries intact (if meaningful on the receiving operating system).

TRANSLATE *filename*
specifies the name of the TCP/IP EBCDIC to ASCII translation table file to be used when transferring this file over a synchronous UFT connection (UFTSYNC or UFTASYNC) or via SMTP (when the MIME option is specified). The file type of the translation table file is determined by the HANGEUL, KANJI, NODBCS, and TCHINESE options. This is used only if UFTSYNC ASCII, UFTSYNC TEXT, UFTASYNC ASCII, UFTASYNC TEXT, MIME ASCII, or MIME ASCII-INLINE is specified. The default file name is STANDARD.

For more information, see *z/VM: TCP/IP User's Guide*.

NODBCS
specifies no DBCS strings are contained in the file data. The file type of the TCP/IP translation table file is TCPXLBIN. This is used only if UFTSYNC ASCII, UFTSYNC TEXT, UFTASYNC ASCII, UFTASYNC TEXT, MIME ASCII, or MIME ASCII-INLINE is specified.

HANGEUL
specifies DBCS strings are contained in the file data. The file type of the TCP/IP translation table file is TCPHGBIN. This is used only if UFTSYNC ASCII, UFTSYNC TEXT, UFTASYNC ASCII, UFTASYNC TEXT, MIME ASCII, or MIME ASCII-INLINE is specified.

KANJI
specifies DBCS strings are contained in the file data. The file type of the TCP/IP translation table file is TCPKJBIN. This is used only if UFTSYNC ASCII, UFTSYNC TEXT, UFTASYNC ASCII, UFTASYNC TEXT, MIME ASCII, or MIME ASCII-INLINE is specified.

TCHINESE
specifies DBCS strings are contained in the file data. The file type of the TCP/IP translation table file is TCPCHBIN. This is used only if UFTSYNC ASCII, UFTSYNC TEXT, UFTASYNC ASCII, UFTASYNC TEXT, MIME ASCII, or MIME ASCII-INLINE is specified.

Usage Notes

1. Tailoring the SENDFILE Command Options

   You can use the DEFAULTS command to set up options and 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. For more information, see “DEFAULTS” on page 153.

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 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 enter a file mode number along with the file mode letter when specifying one file, the file mode number is ignored.

If you want to select the files from a list, you can type an asterisk (*) for file name, file type, or file mode. An asterisk means 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, file type, or both to request 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 list only the files with a particular file mode number, specify the numeric portion of the file mode along with the file mode letter.

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 Enter to send the files.

Note: If you have specified a user ID or node name that contains reserved characters (like '/') on the SENDFILE menu, it will be treated just as if you had specified it directly on the FILELIST screen. This could result in syntax errors. You can bypass this possibility by using a NAMES file.

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 *fileid* passed to the SENDFILE EXEC cannot be a mixed case *fileid*.

The Recipient(s)

You type the name(s) of the recipient(s) in the space provided. For more information on how a name can take the form of a user ID or nickname, see “Operands” on page 890.

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 one of these:

```
File MY DATA A1 sent to JONES at NODE1 on 10/10/81 11:30:25 EDT
File MY DATA A1 sent to JONES at NODE1 on 12/31/1999 11:30:25 EDT
File MY DATA A1 sent to JONES at NODE1 on 2000-02-29 11:30:25 EDT
File MY DATA A1 sent to JONES at VNET.IBM.COM on 1997-09-23 09:33:18
```

  depending on the first valid record in your *userid* NETLOG file.

Note: The date format for the date is the same as the first valid record in your *userid* NETLOG file. If it is a new file, the date format for the date will default to ISODATE (*yyyy-mm-dd*). To change the date formats in your *userid* NETLOG file, see "NETLCNVT" on page 566.

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?

For information on saving a file list, see "FILELIST" on page 291, Usage Note "7" on page 298, "Saving a List of Files". 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.

0

TCP/IP destinations to be transmitted by Unsolicited File Transfer?

Any TCP/IP destinations will default to transmitting the file through SMTP. Type 1 if you want the file to be sent through UFT instead.

The Spool Class

You type the spool class you want used when sending the file. The default spool class is A.

Sending a File

If you specified only one file ID, press either PF5 or Enter after filling out the SENDFILE menu. PF5 sends the file and exits from the menu. Pressing Enter 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. Press Enter 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.)

PF1 Help
Display information about the SENDFILE command.

PF2
Not assigned.

PF3 Quit
Exit from the menu.

PF4
Not assigned.

PF5 Send
Send the file(s) and exit from the menu.

PF6
Not assigned.

PF7
Not assigned.

PF8
Not assigned.

PF9
Not assigned.

PF10
Not assigned.

PF11
Not assigned.

PF12 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.)

PF1 Help
Display information about the FILELIST command.

PF2 Refresh
Update the list to indicate new files, erased files, and so forth, using the same parameters as those specified on the SENDFILE panel.

PF3 Quit
Exit from the list.

PF4 Sort
Files by file type, file name.

PF5 Sendfile
At cursor. Append the fn ft fm on this line and send the file.

PF6 Sort
Files by size, largest first.

PF7 Backward
Scroll backward one screen.

PF8 Forward
Scroll forward one screen.

PF9 FL/n
Issue the command FILELIST /n * * at the cursor, so a list is displayed, containing all files that have the file name displayed on the line with the cursor.

PF10
Not assigned.

PF11 XEDIT
Edit the file pointed to by the cursor.

PF12 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 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).

For an example of a SENDFILE menu, see Figure 45 on page 899. For an example of a FILELIST screen, see Figure 46 on page 899.

3. Format of the File Sent by SENDFILE

   The format of the file sent via RSCS depends on whether the OLD or NEW (the default) option is specified.

   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 sends the file. The OLD option should be used if the recipient does not have the RECEIVE or NETDATA commands 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 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 or NETDATA command on the CMS system, or when the file is being sent to the MVS operating system with TSO Extensions licensed program.

4. Priority

   When SENDFILE is issued with the NEW option to send a file across the network via RSCS (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. The default for SENDFILE when sending via RSCS is to send files as CLASS A NOCONT NOHOLD regardless of the class to which you spool your PUNCH. If you want SENDFILE to use the current PUNCH spool class, specify the CLASS = option on the SENDFILE command. The CP message generated, containing the spool ID, and so forth, 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
```

7. When sending a note, the note cannot be appended to a packed notebook. Before using SENDFILE (or NOTE) to send the note, use COPYFILE with the UNPACK option to change the file format from packed to unpacked.
8. Sending Files to MVS/TSO:

   Using SENDFILE to transfer file mode 4 OS simulated V format files to TSO will generate a file TSO cannot process. You must convert these files to file mode 1 (CMS files) using the FILEDEF and MOVEFILE commands, as in this example:

   ```
   filedef in disk fn ft fm4 (recf v
   filedef out disk fn ft fm1 (recf v
   movefile in out
   ```

   Do not simply rename the file from file mode 4 to file mode 1.

   If the recipient requires the file be in OS simulated format, SENDFILE cannot be used to send the file.
9. When SENDFILE is issued with the NOTE option and the *fm* is specified other than an asterisk, the *fm* must be a read/write disk. If a read only disk is specified, an error message will be issued stating the note file is not found.
10. When the SMTP, MIME, or UFTASYNC option is specified, SENDFILE sends the file to the server identified in the TCP/IP DATA file.
11. When the UFTASYNC option is used, SENDFILE tags the file with a pseudo node of UFT. If you are using RSCS as your UFT server, you must have a LINK or GROUP called UFT.
12. The TCPEXIT EXEC is passed a keyword followed by a parameter list. If you create a TCPEXIT EXEC, it will only be called if an option other than RSCS is specified. The TCPEXIT EXEC is passed the keyword IS_HOST_TCPIP followed by an array name which contains entries consisting of USERID NODE FLAG, where FLAG is a 1 or a 0. This is the flag which tells SENDFILE whether the node is to be treated as a TCP/IP address. A 0 means no, and a 1 means yes. You can place the logic code in TCPEXIT to change these flags to fit your rules. The output array expected should be in the same format as passed in. Currently, the only keyword handled is IS_HOST_TCPIP.
13. When MIME option is specified, the Content-Type header is based on a combination of the file type and specified options.

    When BINARY or BINARY-ATTACH, always use:

    ```
    application/octet-stream
    ```

    When BINARY-INLINE, use:

    ```
    application/postscript for filetypes PS, LISTPS, and PDF
    image/jpeg for filetypes JPG, JPEG, and JPE
    image/bitmap for filetype BMP
    image/gif for filetype GIF
    application/msword for filetype DOC
    application/x-zip for filetype ZIP
    audio/basic for filetypes WAV and AU
    video/mpeg for filetypes MPG and MPEG
    application/octet-stream for all other filetypes
    ```

    When ASCII or ASCII-INLINE, use:

    ```
    text/html for filetypes HTML and HTM
    text/plain for all other filetypes
    ```

    When ASCII-ATTACH, always use:

    ```
    text/plain
    ```
14. In an SSI cluster, if you issue the SENDFILE command without specifying the node, and the recipient is a single-configuration virtual machine, the recipient can display or receive the file when logged on

to any member of the cluster. If you omit the node and the recipient is a multiconfiguration virtual machine, the recipient can display or receive the file only when logged on to the member where the SENDFILE command was issued. If you specify the node for either type of user, the file will be sent only if RSCS is running on both members.

Examples

The sender entered SF to get the screen shown in Figure 45 on page 899. The sender then 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, shown in Figure 46 on page 899. The files to be sent can be selected from this screen.

```
                ----------------  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?
    0    TCP/IP destinations to be transmitted by Unsolicited File Transfer ?



    A    Spool class to use when sending the file(s)


   1= Help            3= Quit             5= Send             12= Cursor

====>
                                                          Macro-read 1 File
```

Figure 45. Sample SENDFILE Menu

```
 SNOWHITE FILELIST A0  V 108  Trunc=108 Size=418 Line=1 Col=1 Alt=0
Cmd   Filename Filetype Fm Format Lrecl    Records    Blocks   Date     Time
      WISTFUL  DATA     A1 V         95         34         2 10/04/88 21:12:04
 s    BOSS     DATA     A1 V         95         29         2 10/04/88 20:58:07
      DUMMY    DATA     A1 V        107        281        10 10/04/88 17:59:00
 s    GROUCHY  DATA     A1 V         92        101         4 10/02/88 15:33:05
      PRINCE   DATA     A2 V         75         28         1  9/25/88 12:10:03
 s    SNOOZY   DATA     A2 V        120        277        10  9/24/88  9:14:02
      SNIFFLES DATA     A1 V         26          7         1  9/23/88 16:50:06
      WITCH    DATA     A1 V         80        489        30  8/26/88 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 46. 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 send, as shown in Figure 46 on page 899, then press Enter. You can also 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

- DMS002E File *fn ft fm* not found [RC=28]
- DMS006E No read/write filemode accessed [RC=36]
- DMS048E Invalid filemode *mode* [RC=24]
- DMS054E Incomplete fileid specified [RC=24]
- DMS060I File TCPIP DATA not found; domain name could not be determined
- DMS062E Invalid character * in fileid *fn ft fm* [RC=20]
- DMS069E Filemode *mode* not accessed [RC=36]
- DMS081E Invalid reply; {answer 1 for YES and 0 for NO|enter a valid spool class}
- DMS149E Userid *userid* not valid; no files have been sent
- DMS399E Tag too long for *nickname* in *userid* NAMES file [RC=88]
- DMS579E Records truncated to *nn* when added to *fn ft fm* [RC=3]
- DMS637E Missing nodeid for the AT operand [RC=24]
- DMS647E Userid not specified for *nickname* in *userid* NAMES file [RC=32]
- DMS648E Userid *name* not {found|resolved}; no files have been sent [RC=32]
- DMS653E Error executing *command*, rc=*rc* [RC=40]
- DMS657E Undefined PFkey/PAkey
- DMS667E NOTE header does not contain the {keyword From:|keyword To:|OPTIONS line|DATE line} [RC=32]
- DMS671E Error sending file *fn ft fm*; rc=*nn* from *command* [RC=100]
- DMS672E Virtual punch invalid or not defined [RC=36]
- DMS673E Addressees are in the note header cards; do not specify names with NOTE option [RC=24]
- DMS674E Punch is not ready [RC=36]
- DMS675E No names specified [RC=24]
- DMS676E Invalid character * for Network ID [RC=20]
- DMS677E Invalid option: *option* in option line [RC=32]
- DMS678E Invalid note header format; note cannot be sent [RC=32]
- DMS679E Filemode *mode* is not accessed; note cannot be sent [RC=36]
- DMS680E Invalid fileid specified with FILELIST option [RC=20]
- DMS743E File *fn ft fm* is in an invalid format [RC=40]
- DMS743E Note not appended to notebook. RC=*nn* from command [RC=*nn*]
- DMS1012E Node ID *node* not valid for RSCS; no files have been sent [RC=32]

- DMS1030E IPv6 addresses cannot be used with the {UFTSYNC/UFTASYNC} option; no files have been sent.
- DMS1116E Invalid value *value* for {Hostname|DomainOrigin} in TCPIP DATA [RC=32]
- DMS2501E One or more lines between the {OPTIONS:|USEROPTIONS:} line and the DATE: line contain non-blank characters.
- DMS2548E UFT serverID not defined in TCP/IP; no files have been sent [RC=53]
- DMS3279E Remote server response: response

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in the Shared File System	“File Pool Server Messages” on page 1414
Errors in command syntax	“Command Syntax Error Messages” on page 1411

SENTRIES

Authorization

General User

Purpose

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.

Usage Notes

If you issue a SENTRIES command in an exec that has set up a procedure to be executed when an error occurs, a nonzero SENTRIES return code causes that procedure to execute.

SET

Authorization

General User

Purpose

Use the SET command to establish, turn off, or reset a particular function in your CMS virtual machine. Only one operand may be specified per SET command. SET cannot be issued without an operand.

Operands

The operands available with SET are summarized below.

ABBREV	GETMAIN	RECALL
APL	IMESCAPE	REDTYPE
AUTODUMP	IMPCP	RELPAGE
AUTOREAD	IMPEX	REMOTE
BLIP	INPUT	RESERVED
BORDER	INSTSEG	RORESpect
CHARMODE	KEYPROTect	SERVER
CMSPF	LANGuage	STORECLR
CMSTYPE	LDRTBLS	SYSNAME
CMS370AC	LINEND	TAPECSL
COMDIR	LOADAREA	TAPENEVR
DOS	LOCATION	TEXT
DOSLNCNT	LOGFILE	THReshold
DOSPART	MACLsubs	TRANslate
EXECTRACe	NONDISP	TRAPMSG
FILEPool	NONSHARE	TVICALL
FILESPace	OLDCMDS	UPSI
FILEWait	OSTXTBUF	VSCREEN
FULLREAD	OUTPUT	WINDOW
FULLSCREen	PROTECT	WMPF
GEN370	RDYMSG	

Usage Notes

General Usage Notes for all SET command Operands

1. If you issue the SET command specifying a not valid function and the implied CP function is in effect, you may receive message HCPCFC003E Invalid option - *option*.
2. If a not valid SET command function is specified from an exec and the implied CP function is in effect, 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 their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411
Errors in the Shared File System	“File Pool Server Messages” on page 1414

SET ABBREV

Authorization

General User

Purpose

Use the SET ABBREV command 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.

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, 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 more information on the relationship of the SET ABBREV and SYNONYM commands, see "SYNONYM" on page 1056.

Initial Setting

ABBREV ON

Messages and Return Codes

- DMS109S Virtual storage capacity exceeded [RC=104]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	"Command Syntax Error Messages" on page 1411

SET APL

Authorization

General User

Purpose

Use the SET APL command to start or stop character code conversion for APL characters for XEDIT and CMS.

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 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 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 XEDIT, and changing the APL setting for XEDIT also changes the APL setting for CMS.

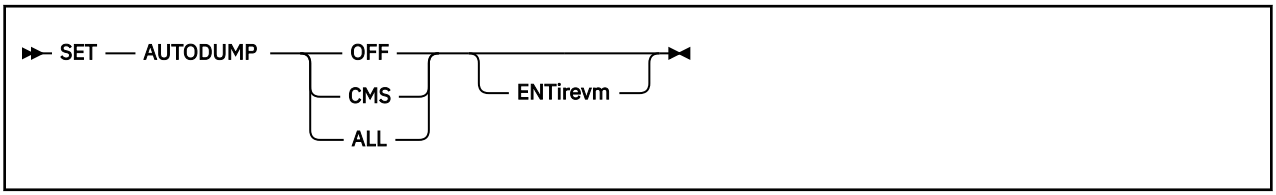
Messages and Return Codes

- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS329W Warning: APL/TEXT option not in effect
- DMS524W NONDISP character reset to ".
- DMS926E 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 their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

SET AUTODUMP



Authorization

General User

Purpose

Use the SET AUTODUMP command to control the type of dump created by CMS in case of an abend, and when an automatic dump of the virtual machine will occur.

You can create a dump for irrecoverable CMS system abends, for all abends that occur in your virtual machine, or you can turn off the automatic dump creation.

You can choose to dump part of, or the entire, virtual machine.

Operands

OFF
indicates you do not want a dump for any abends. This is the initial setting.

CMS
indicates you want to take a dump whenever an irrecoverable CMS system abend occurs and CMS enters a disabled wait state.

ALL
indicates you want to take a dump for all abends within the virtual machine, both recoverable and irrecoverable.

ENTirevm
when specified with ALL or CMS, indicates automatic dumps will contain the following:

- The entire virtual machine
- Any saved segments the machine is currently using

In addition to the dump, any data spaces currently in use that contain SFS data are dumped (in VMDUMP format).

If ENTIREVM is *not* specified, the default dump will contain:

- The DMSNUC region of CMS
- The storage management work area
- The page allocation table
- The loader tables

Note: A dump is not produced by the HX command, or when a program check is trapped using STAE, SPIE, or ABNEXIT.

Initial Setting

AUTODUMP OFF

Usage Notes

1. If you specify SET AUTODUMP CMS, a dump will be taken for the following irrecoverable CMS system errors:
   - Program checks within nucleus resident modules
   - Irrecoverable errors in the file system
   - Irrecoverable storage management errors
   - Errors that result in a disabled wait PSW
2. For some severe irrecoverable file system errors detected by CMS, an entire virtual machine dump will be generated unless SET AUTODUMP OFF has been specified.
3. If you specify SET AUTODUMP ALL, a dump will be taken for the errors mentioned in Usage Note "1" on page 908, as well as these conditions:
   - All program checks
   - Use of the ABEND macro
   - Use of the DMSABN macro
4. If you have set AUTODUMP to ALL or CMS, the dump is produced using the CP VMDUMP facility. The dump is sent to the reader of the virtual machine where the abend occurred.
5. You can use the DUMPLOAD utility to process the dump, and the DUMPSCAN command (CMSPOINT subcommand) of the Dump Viewing Facility to view it. For more information on the DUMPLOAD utility, see *z/VM: CP Commands and Utilities Reference*. For more information on the Dump Viewing Facility and the DUMPSCAN command, see *z/VM: Dump Viewing Facility*.

Responses

When the system produces a dump, you will receive the messages:

```
'DMSABE2047I AUTODUMP dump started; please wait'
```

If the dump was successful:

```
'DMSABE1297I Dump has been taken'
```

If the dump was unsuccessful:

```
'DMSABE1297I Dump failed; condition code = cc; return code = rc'
```

If ENTIREVM was specified and there is at least one SFS data space in the virtual machine, you will also receive the following messages:

```
'DMSDDS2047I AUTODUMP dump started for data space:
ASIT  = xxxxxxxxxxxxxx; please wait'
```

If the dump was successful:

```
'DMSABE1297I Dump has been taken'
```

If the dump was unsuccessful:

```
'DMSABE1297I Dump failed; condition code = cc; return code = rc'
```

For the list of return codes and reason codes, see the DIAGNOSE X'94' in *z/VM: CP Programming Services*.

Messages and Return Codes

Return codes:

RC
Meaning

0
Normal completion

24
A not valid parameter was specified for the SET AUTODUMP command

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

SET AUTOREAD

Authorization

General User

Purpose

Use the SET AUTOREAD command to specify whether a console read is to be issued after command execution.

Operands

ON
specifies a console read is to be issued immediately after command execution. ON is the initial setting for nondisplay, nonbuffered terminals.

OFF
specifies you do not want a console read to be issued until you press Enter 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 OFF for display terminals.

AUTOREAD ON for nondisplay, nonbuffered terminals.

Usage Notes

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

- DMS109S Virtual storage capacity exceeded [RC=104]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

SET BLIP

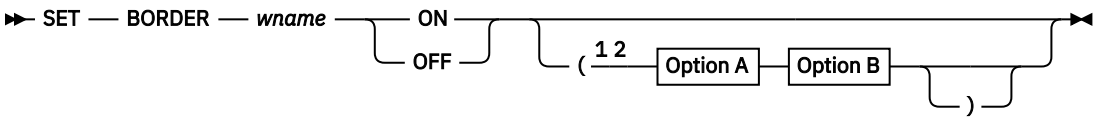
Authorization

General User

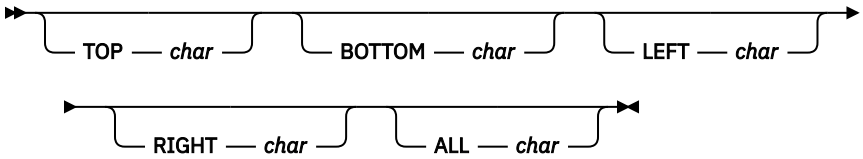
Purpose

If you issue the SET BLIP command, CMS ignores the operand and the BLIP setting remains OFF. CMS does not support the BLIP facility. This command is maintained for compatibility purposes only.

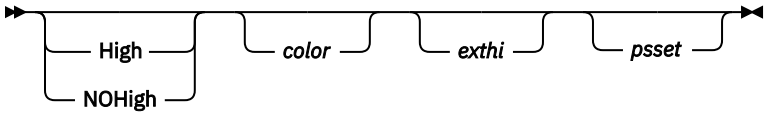
SET BORDER



Option A



Option B



Notes:

[1] You can enter Options in any order between the parentheses.
[2] Only those window border edges being referred to by the options in option A are set to ON. All other border edges are set to OFF.

Authorization

General User

Purpose

Use the SET BORDER command to define borders around windows. Borders visually separate information displayed in different windows. Border corners can also be used to enter Border Commands, which manipulate windows. For more information, see “Window Border Commands” on page 1251.

Operands

wname
is the name of the window.

ON
indicates 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 the borders will not be displayed.

Options

Option A

allows you to specify new characters for window borders. One or more of the following options may be specified:

TOP *char*
where *char* specifies the character that is displayed in the top border.

BOTTOM *char*
where *char* specifies the character that is displayed in the bottom border.

LEFT *char*
where *char* specifies the character that is displayed in the left-hand border.

RIGHT *char*
where *char* specifies the character that is displayed in the right-hand border.

ALL *char*
where *char* specifies the character that is displayed in all borders.

Option B

indicates how the borders should be displayed. These 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 programmed symbol set must be loaded in the display to be used. If not, the default PSset is used.

Initial Setting

The initial setting is determined by the WINDOW DEFINE command.

Usage Notes

1. To override the default border characteristics see the WINDOW DEFINE command 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 to ON. All other edges are set to OFF. For more information, see “WINDOW DEFINE” on page 1220.

   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 any or all 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 “Window Border Commands” on page 1251. All other border characters are protected. The corner character, ‘+’, may be displayed as a different character depending on the programmed symbol set used to display the border.
4. SET CHARMODE must be ON to display border characters using programmed symbol set 1 (PS1).

5. 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

- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS639E Error in NUCXLOAD routine; return code was *nnn* [RC=24]
- DMS921E Window *wname* is not defined [RC=28]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

SET CHARMODE

Authorization

General User

Purpose

Use the SET CHARMODE command to specify whether the character attributes or field attributes should be used when displaying virtual screen data on the physical screen.

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. For more information on how to specify character attributes when writing data, see "VSCREEN WRITE" on page 1192.
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), field attributes are used regardless of the CHARMODE setting. For more information, see *IBM 3270 Information Display System Data Stream Programmer's Reference*.
3. SET CHARMODE must be ON to update COLOR, EXTHI, and PSS in the VSCREEN WRITE 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 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 VSCREEN WRITE command within a DBCS string.

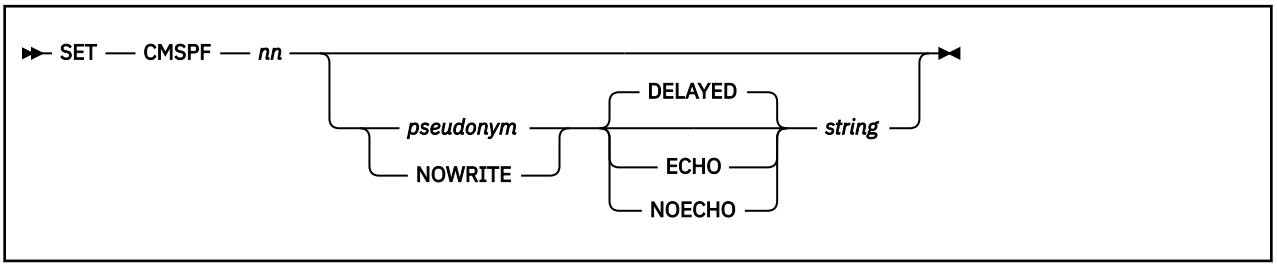
Messages and Return Codes

- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS639E Error in NUCXLOAD routine; return code was *nnn* [RC=24]
- DMS926E 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 their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

SET CMSPF



Authorization

General User

Purpose

Use the SET CMSPF command to set a CMSPF key to a specific command. The CMSPF keys are used when SET FULLSCREEN is ON.

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. Mixed DBCS data can be used for *pseudonym*, provided your terminal is capable of supporting mixed DBCS.

NOWRITE
suppresses overwriting of the PF key pseudonym when you set a CMSPF key.

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 Enter. 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 a string, or command(s) to be executed when the key is pressed.

Initial Setting

The initial settings for the CMSPF keys are as follows:

```
CMSPF 01 Help       ECHO    HELP
CMSPF 02 Pop_Msg    NOECHO  WINDOW  POP MESSAGE *
CMSPF 03 Quit       NOECHO  SET FULLSCREEN SUSPEND
CMSPF 04 Clear_Top  NOECHO  #WM WINDOW CLEAR =
```

```
CMSPF 05 Filelist   ECHO    EXEC FILELIST
CMSPF 06 Retrieve           RETRIEVE
CMSPF 07 Backward   NOECHO  #WM WINDOW BACKWARD CMS 1
CMSPF 08 Forward    NOECHO  #WM WINDOW FORWARD CMS 1
CMSPF 09 Rdrlist    ECHO    EXEC RDRLIST
CMSPF 10 Left       NOECHO  #WM WINDOW LEFT CMS 10
CMSPF 11 Right      NOECHO  #WM WINDOW RIGHT CMS 10
CMSPF 12 Cmdline    NOECHO  VSCREEN CURSOR 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 to 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 (resetting it to no operation), 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, 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.

Messages and Return Codes

- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS525E Invalid PFkey number [RC=24]
- DMS639E Error in NUCXLOAD routine; return code was *nnn* [RC=24]
- DMS926E Command is only valid in CMS FULLSCREEN mode [RC=88]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

SET CMSTYPE

Authorization

General User

Purpose

Use the SET CMSTYPE command to suppress or resume CMS terminal display within an exec.

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 (for example, 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 full-screen 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

System messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	"Command Syntax Error Messages" on page 1411

SET CMS370AC

Authorization

General User

Purpose

Use the SET CMS370AC command to turn CMS support on or off for System/370 applications that will not execute properly in ESA, XA, and XC virtual machines.

Operands

ON
Turns the CMS370AC facility ON to help support System/370 applications that will not execute properly in ESA, XA, and XC virtual machines. In addition, if CP SET 370ACCOM is OFF, issuing SET CMS370AC ON will turn CP SET 370ACCOM to ON.

OFF
Turns the CMS370AC facility OFF. In addition, if the SET CMS370AC ON command turned the CP SET 370ACCOM to ON, it will be turned OFF when SET CMS370AC is turned OFF.

Initial Setting

OFF

Usage Notes

1. It is recommended you first try running your application with SET CP 370ACCOM turned ON. If the application does not execute successfully, you should issue SET CMS370AC ON and try running your application again.
2. If your application requires you issue SET CMS370AC ON, the command must be issued before executing any applications that may manipulate the I/O and External New PSWs. Failure to do so may cause unpredictable results.
3. CP SET 370ACCOM must not be turned OFF while SET CMS370AC is set ON. Results are unpredictable, but you may see messages similar to these:

```
DMSITP143T  Specification exception occurred at
            hhhhhhhh in routine cccccccc;
            re-IPL CMS
HCPGIR450W  CP entered; disabled wait PSW
            hhhhhhhh hhhhhhhh
```

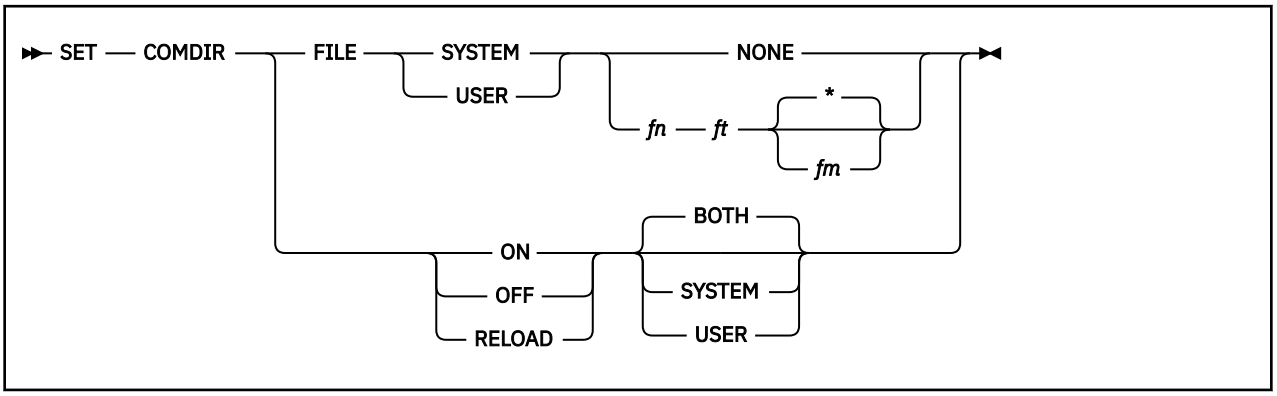
4. If CP SET 370ACCOM was set ON prior to invoking SET CMS370AC ON, issuing SET CMS370AC OFF will not turn CP SET 370ACCOM off.
5. The SET CMS370AC command is not supported by z/Architecture CMS.

For more information regarding the CP SET 370ACCOM command and the CP 370 Accommodation Facility, see *z/VM: CP Programming Services* and *z/VM: CP Commands and Utilities Reference*.

Messages and Return Codes

- DMS2632E A virtual machine in z/Architecture mode may not have CMS370AC on [RC=24]

SET COMDIR



Authorization

General User

Purpose

Use the SET COMDIR command to set up and control your CMS communications directories. For more information on CMS communications directories, see *z/VM: CMS Application Development Guide*.

Operands

FILE
sets the file name of the directory.

ON
turns on name resolution.

OFF
turns off name resolution.

RELOAD
causes the storage-resident static copy of the communications directory (or directories) to be reloaded.

Note: Changing the disk-resident copy of a directory does not change the storage resident copy; SET COMDIR FILE or RELOAD must be issued to reload the file.

SYSTEM
USER
BOTH
indicates the directory level to set. If you specify BOTH, the SET COMDIR function you specify applies to both user and system directories. If only one is loaded, it applies to that one. BOTH is the default for the ON, OFF, and RELOAD operands.

If you specify SET COMDIR RELOAD BOTH, the IPC names file, $QUEUES$ NAMES, is loaded along with the system and user directories. Additionally, specifying SET COMDIR RELOAD BOTH loads the IBM-level communication directory file. This file contains communications directory entries that are provided for use by IBM. For more information on using this combination of options, see *z/VM: CMS Application Multitasking*.

NONE
indicates 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.

fn ft fm
specifies the file to associate with the specified directory level and loads the file into storage.

Usage Notes

1. There can be three levels of communications directory files: a SYSTEM level, a USER level, and an IBM level.

   A *system-level* communications directory should be set up by a system administrator. You can also set up a *user-level* communications directory for your own use, to include name definitions not contained in the system file. The *IBM-level* communications directory is installed by IBM and contains those communication directory entries used by IBM. The name of the IBM-level file is ICOMDIR NAMES.
2. The NAMES command invoked with the "COMDIR" option can be used to create and maintain your communications directory files. In order to use the NAMES command to update the file, the file type of your communications directory file must be "NAMES" and the file mode must be "*".

   If the file type is not "NAMES" or the file mode is not "*", you must use an editor to create and maintain your communications directory files. For more information on how to use tags in your names file to set up a user communications directory, see “NAMEFIND” on page 516.
3. When you enter SET COMDIR FILE, CMS immediately tries to find the specified file and load it into storage. For the SET to take effect, the specified file must be present at the time the SET is entered. If the specified file is not found or cannot be read, CMS leaves the specified Communications Directory level unchanged.
4. If your installation has not modified your SYSPROF EXEC, CMS issues these SET COMDIR commands:

   ```
   SET COMDIR FILE SYSTEM SCOMDIR NAMES *
   SET COMDIR FILE USER UCOMDIR NAMES *
   SET COMDIR RELOAD
   ```

   CMS attempts the system-level SET with only the S-disk accessed and the user-level SET with only the A-disk and S-disk accessed. Therefore, if you are attempting to use the user-level Communications Directory, you must either put a UCOMDIR NAMES file on your A-disk or enter the appropriate SET COMDIR FILE command after you access the minidisk or SFS directory containing your Communications Directory file. These ACCESS and SET COMDIR FILE commands can be placed in your PROFILE EXEC if you like.

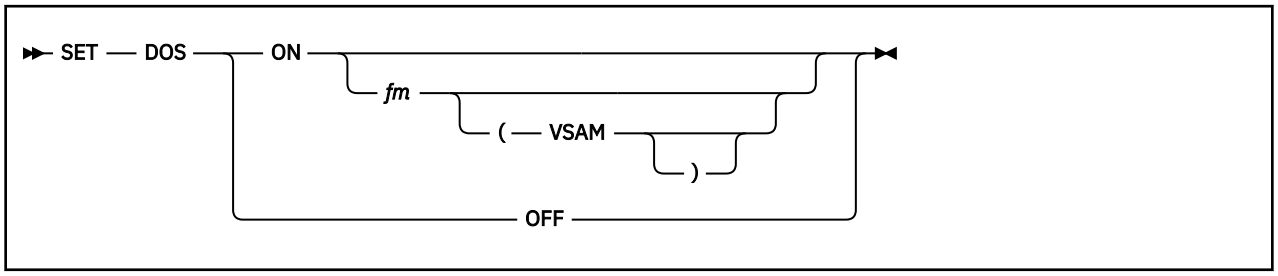
Messages and Return Codes

- DMS639E Error in routine *routine*; return code was *nnn* [RC=*rc*]
- DMS1286E Error loading {SYSTEM|USER} Communications Directory, fileid = *fn ft fm* [RC=4|8|12]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411
Errors in the Shared File System	“File Pool Server Messages” on page 1414

SET DOS



Authorization

General User

Purpose

Use the SET DOS command 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.
- You are going to use the AMSERV command or you are going to execute programs to access VSAM data sets.

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 licensed program and supports neither VSE/SP 2.1 and later nor VSE/ESA libraries.

VSAM
specifies 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

Usage Notes

When DOS is set to ON, message DMS1101I is issued to tell you where the DOS partition is located and how much space was obtained. It is only issued when the command is called by being typed at the terminal, by the CMDCALL command from EXEC 2, or by the ADDRESS CMS instruction from REXX.

Messages and Return Codes

- DMS048E Invalid filemode *mode* [RC=24]
- DMS109S Virtual storage capacity exceeded [RC=104]

- DMS333E *nnnnn*K bytes of contiguous free storage are not available to establish the DOS partition at location 20000 [RC=24]
- DMS402W DMSLBR not in CMSBAM segment; ESERV support not available [RC=104]
- DMS410S Control program error indication *xxx* [RC=*rc*]
- DMS444E Volume *volid* is not a DOS SYSRES [RC=32]
- DMS804E Error establishing CMS/DOS environment [RC=*nn*]
- DMS1101I *nnnnn*K DOS partition defined at hexadecimal location *xxxxxxxx*

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

Note: In RC=*rc*, the *rc* represents the actual error code generated by CP.

SET DOSLNCNT

Authorization

General User

Purpose

Use the SET DOSLNCNT command to specify the number of SYSLST lines per page for the CMS/DOS environment.

Operands

DOSLNCNT *nn*
specifies the number of SYSLST lines per page. The *nn* is an integer from 30 to 99.

Usage Notes

If the value of the LPP option on the CP SPOOL command is 0 (LPP OFF), the default DOSLNCNT setting when you SET DOS ON will be 56. If the LPP value is greater than 0, the default DOSLNCNT setting when you SET DOS ON will equal the LPP value. If the LPP value is greater than the maximum DOSLNCNT value of 99, the DOSLNCNT value will be set to 99.

Issuing SET DOSLNCNT will override the fault setting. For more information, see *z/VM: CP Commands and Utilities Reference*.

Initial Setting

056

Messages and Return Codes

- DMS099E CMS/DOS environment not active [RC=40]
- DMS109S Virtual storage capacity exceeded [RC=104]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

SET DOSPART

Authorization

General User

Purpose

Use the SET DOSPART command to specify the size of the partition in which you want a program to execute in the CMS/DOS environment.

Note: The DOS partition is established at LOCATION 20000.

Operands

***nnnnn*K**
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.

Note: The size is defined on a 4KB boundary.

You should use this function when you can control the performance of a particular program by reducing the amount of available virtual storage; or, if the default size, which is usually 100KB, is not large enough to contain the program to be run. (The default is established when DOS is set to ON with the SET DOS ON command.) When this happens you can either let CMS attempt to get additional storage contiguous to the end of the partition or you can use the SET DOSPART *nnnnn*K command to redefine the size of the partition.

Note: The CMS/DOS partition, unlike partitions under DOS, is used only to load and run programs started by the FETCH or LOAD command. Areas allocated by GETVIS are assigned addresses outside the partition but within the user's virtual machine.

OFF
specifies you no longer want to control your virtual machine partition size. When the DOSPART setting is OFF, the storage allocated to the partition is redefined to the default size. CMS tries to allocate the default value of 100KB of contiguous storage. If 100KB is not available, it tries to allocate storage down to a minimum of 20KB.

Initial Setting

DOSPART OFF

Usage Notes

1. When a DOS partition is defined, message DMS1101I is issued to tell you how much space was obtained and where the partition is located.

   Note: It is issued only when you:

   - Enter the command at the terminal
   - Issue it using the CMDCALL command from an EXEC 2 exec
   - Issue it using the ADDRESS CMS instruction from a REXX exec
2. You can issue QUERY DOSPART to display the current setting of the virtual partition size.

3. For SET DOSPART OFF, a default-size partition is defined. If the virtual machine requires more than the default-size partition, use the SET DOSPART *nnnnn*K command to define a larger partition.
4. Error message DMS333E is given only if DOS is unable to get the amount of storage it needs. For more information on getting more storage, see the CP DEFINE command in *z/VM: CP Commands and Utilities Reference*.

Messages and Return Codes

- DMS099E CMS/DOS environment not active [RC=40]
- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS333E *nnnnn*K bytes of contiguous free storage are not available to establish the DOS partition at location 20000 [RC=24]
- DMS1101I *nnnnn*K DOS partition defined at hexadecimal location *xxxxxx* [RC=0]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

SET EXECTRACE

Authorization

General User

Purpose

Use the SET EXECTRAC command to set tracing on or off for your REXX or EXEC 2 program.

Operands

ON
specifies you want tracing turned on for your REXX or EXEC 2 program. Upon return to CMS or XEDIT, tracing is turned off.

OFF
specifies you want tracing turned off for your REXX or EXEC 2 program.

Initial Setting

EXECTRAC OFF

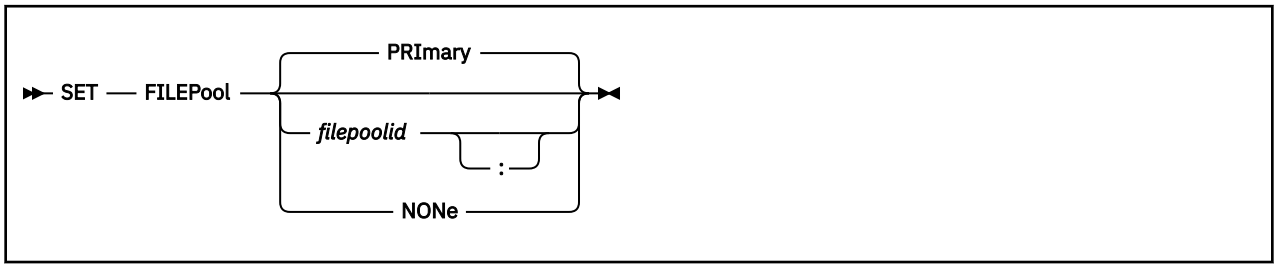
Messages and Return Codes

- DMS109S Virtual storage capacity exceeded [RC=104]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

SET FILEPOOL



Authorization

General User

Purpose

Use the SET FILEPOOL command 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.

Operands

PRImary
resets the default file pool to the value of the FILEPOOL keyword on the IPL statement when you began your CMS session. If the FILEPOOL keyword was not specified on the IPL statement, the default file pool is reset to NONE.

filepoolid
filepoolid:
is the name of the file pool you want as the default file pool. The colon is optional.

NONe
specifies you do not want a default file pool.

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
     ```

   The default file pool can be defined after IPL by:
   - Entering the SET FILEPOOL command from the command line.
   - Placing

     ```
     SET FILEPOOL filepoolid
     ```

     in your PROFILE EXEC if your 191 disk is accessed as file mode A.

If the default file pool is set after IPL, a subsequent SET FILEPOOL PRIMARY command will reset the default file pool to NONE. The primary file pool is the file pool defined on the IPL statement by the value of the FILEPOOL keyword.

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.

Messages and Return Codes

- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS389E Invalid operand: *operand* [RC=24]
- DMS391E Unexpected operand(s): *operand* [RC=24]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

SET FILESPACE

Authorization

General User

Purpose

Use the SET FILESPACE command to set or reset your default file space ID. The default file space ID will be used when you do not specify a specific file space ID in a command or CSL routine call. This default is in effect until the CMS session ends. This setting is similar to setting the default file pool ID in the SET FILEPOOL command.

Operands

userid
is the name of the file space you want as the default file space. If *userid* is omitted, the default file space ID is set to the virtual machine ID.

Do not SET FILESPACE to a user ID that begins with a plus (+) or a minus (-) or that contains a colon (:) or a period (.). These characters are used as separator characters in directory IDs of which the user ID is a part.

Usage Notes

1. The default file space can be defined after IPL by:
   - Entering the SET FILESPACE command from the command line.
   - Placing

     ```
     SET FILESPACE userid
     ```

     in your PROFILE EXEC.

     If the default file space is set after IPL, a subsequent SET FILESPACE command with no operands will reset the default file space to the virtual machine ID.
2. The initial setting of the default file space ID is set to the virtual machine.

Examples

If you enter:

```
set filespace farrells
```

and then issue the following ACCESS command:

```
access vmsysu:. b
```

CMS will substitute 'farrells' for the default file space ID. You would have effectively entered:

```
access vmsysu:farrells. b
```

Messages and Return Codes

- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS389E Invalid operand: *operand* [RC=24]
- DMS391E Unexpected operand(s): *operand* [RC=24]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

SET FILEWAIT

Authorization

General User

Purpose

Use the SET FILEWAIT command to indicate whether you want a request for control of an SFS directory, a file in an SFS directory, or a BFS regular file to wait until it becomes available. The conditions under which SFS will apply the FILEWAIT setting are outlined in "Usage Notes" on page 933.

Operands

ON
indicates you do want to wait until the requested SFS object becomes available. If you reference a file or directory, implicit locking will wait until the file or directory becomes available. If you explicitly request to lock an SFS object (storage group, file space, directory, or file) the request will wait on implicit locks until the object is available. You can request explicit locks using CSL routines and commands such as CREATE LOCK, FILEPOOL DISABLE, and XEDIT. The wait may be for a prolonged period of time.

OFF
indicates you do not want to wait until the requested SFS object becomes available. The request fails immediately if the SFS object 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 the availability of an SFS object if the availability is prevented by a lock created by the CREATE LOCK or FILEPOOL DISABLE commands or corresponding CSL routines. XEDIT uses the CREATE LOCK command for existing SFS files in the file control directories, unless the NOLOCK option was specified.
3. Not all read/write access attempts of DIRCONTROL directories will wait when FILEWAIT is ON. If you try to access a DIRCONTROL directory in R/W status, and another user already has the directory accessed R/W, your access attempt does not wait. Instead, the attempt fails (unless you own the directory and did not specify the FORCERW option, in which case you get a read-only access).

   When another user has a file in the DIRCONTROL directory open for writing but does not have the directory accessed, your access attempt will wait. When the user closes the file, you will get R/W access. In this case, you will wait even if you own the directory (you will not get a read-only access).
4. 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.
5. With FILEWAIT ON, if you want to write to a file another user has open to write, your request will wait until the implicit lock is deleted.

6. 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.
7. With FILEWAIT ON and DFSMS/VM active, requests that modify a file that is currently being migrated or recalled from the DFSMS/VM storage repository but do not reference the existing file data (for example, ERASE, COPYFILE with the REPLACE option, or CREATE ALIAS) will wait until the migrate or recall operation completes.
8. For the byte file system, the SET FILEWAIT setting affects those requests entered while in the CMS environment using traditional file and directory IDs (*fn ft fm* or *dirid*). That is, it affects requests entered for byte file system files using CSL routines or CMS commands. It does not affect requests entered from the OPENVM environment through BFS or POSIX interfaces using path name.

Examples

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.

Messages and Return Codes

System messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

SET FULLREAD

Authorization

General User

Purpose

Use the SET FULLREAD command to allow 3270 null characters to be recognized in the middle of the physical screen by CMS and XEDIT.

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 processor, 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 nonfunctional.
5. Changing the FULLREAD setting for CMS also changes the FULLREAD setting for XEDIT.

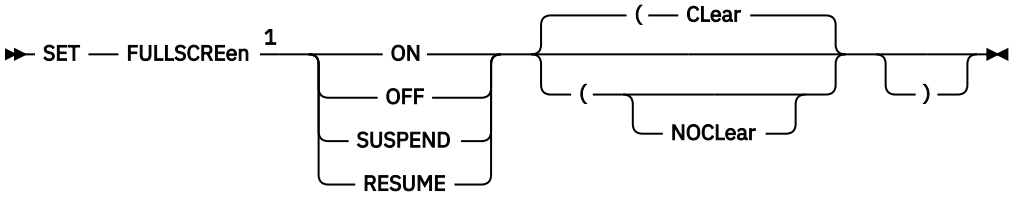
Messages and Return Codes

- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS639E Error in NUCXLOAD routine; return code was *nnn* [RC=24]
- DMS926E 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 their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

SET FULLSCREEN



Notes:

[1] A special command that can be used in the FULLSCREEN environment is #WM.

Authorization

General User

Purpose

Use the SET FULLSCREEN command to control whether CMS uses windowing support (virtual screens and windows) for command input and output, messages, and warnings.

Operands

ON
initializes full-screen CMS. SET FULLSCREEN ON defines the default virtual screens and windows for CMS. Output that is generally displayed by CP is trapped by CMS (by the IUCV Message All System Service) so messages and VM output are displayed in windows.

OFF
returns CMS to line mode operation.

SUSPEND
specifies CMS should temporarily return to line mode operation. CMS discontinues trapping I/O (by severing the *MSGALL connection) so 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 user-defined windows and features (for example: CMSPF Keys, Command Line, Status Area, and so forth) being displayed.
2. When entering full-screen CMS:
   - All default virtual screens 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 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 SET FULLSCREEN is ON, private resource managers are not invoked. 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.

   For more information, see *z/VM: Connectivity*.
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 these commands: WINDOW CLEAR, VSCREEN CLEAR, WINDOW SHOW, or WINDOW HIDE, allows the virtual screen to be updated. That is, the oldest information that has been scrolled is deleted off the top so 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 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 before 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 initial setting), use the CP TERMINAL command.
8. Commands can be entered only 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 WINDOW POP STATUS command.
10. The WM window is automatically displayed in full-screen CMS when the window or windows showing the active virtual screen are not visible on the screen. (The active virtual screen is the one in which a command or response is requested.) For example, you may maximize a window that is not showing the active virtual screen so it covers the window or windows showing the active virtual screen. Because you cannot enter commands in the active virtual screen, the WM environment is automatically entered; the WMPF keys and WM command line are available to manipulate all windows. You can choose to press the PF3 key (if the setting of WMPF key 3 has not been changed from the initial value) or issue the command WINDOW DROP WM to pop all windows showing the active virtual screen and exit the WM environment.

    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 one of the following occurs:

    - The CMS virtual screen is updated.
    - 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 WINDOW DROP WM command (default WMPF 3) returns you to the application.
11. SET FULLSCREEN must be ON in order for CMS to recognize double-byte character set (DBCS) strings, and for them to be displayed. XEDIT also recognizes DBCS characters. For more information, see *z/VM: XEDIT Commands and Macros Reference*.

System Defaults for Full-Screen CMS

1. The default windows for full-screen CMS are:

   Table 40. 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

Table 40. Default Windows (continued)

Wname	Lines	Cols	Psline	Pscol	Options
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

Where:

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 the number of lines in the window may vary depending on the amount of scrollable data displayed.

Fixed
indicates the number of lines in the window is always constant.

Border
indicates 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 borders are not displayed.

Pop
specifies the window is displayed on top of all other windows when the virtual screen the window is showing is updated.

Nopop
specifies there is no effect on the window's position in the ordered list of windows when the virtual screen the window is showing is updated.

Top
specifies the window may qualify as the topmost window.

Notop
specifies 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 WINDOW POP WM, when you press the PA1 key when CMS is the active virtual screen, or when the WM window is automatically displayed.

All default windows are SYSTEM windows, which means the window is retained when the system abnormally ends (abends) or when the HX (halt execution) command is issued.

2. The default virtual screens for full-screen CMS are:

Table 41. Default Virtual Screens

Vname	Lines	Cols	Rtop	Rbot	Color	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

Where:

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
the number of top reserved lines

Rbot
the number of bottom reserved lines

Color
data color

Protected
if protected, you cannot type into the window(s) connected to the virtual screen

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 WINDOW POP 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 this manner:

Table 42. Default Windows and Virtual Screens

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:

   Table 43. Default Settings for Message Routing

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

   Where:

   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.

   For more information on changing the default message routing, see "VSCREEN ROUTE" on page 1184.

   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 when CMS is the active virtual screen displays the WM window. The PA2 key and Clear key scroll the topmost window forward. For more information on the default settings for the CMSPF keys, see "SET CMSPF" on page 917.

   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. Before 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 Clear to scroll forward and refresh the screen.

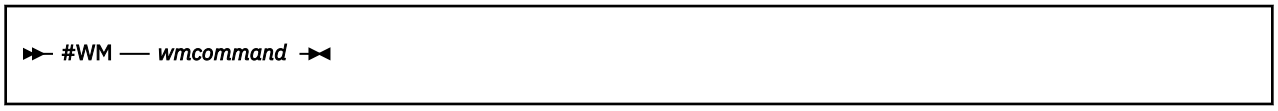
   Alternatively, issue the SET FULLSCREEN SUSPEND command, run the application, and then issue SET FULLSCREEN RESUME when the application completes.
3. SET FULLSCREEN SUSPEND and SET FULLSCREEN RESUME can be *nested*. For example, suppose full-screen CMS is ON and an application called MYPROG issues SET 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.
4. If full-screen CMS has never been set to 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 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. Before 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.
10. SET FULLSCREEN controls only the use of windowing by CMS. Whether FULLSCREEN is ON or OFF, you may define and use your own virtual screens and windows.

#WM - A 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. Use the #WM command to execute a command immediately from the CMS virtual screen.



#WM operands:

wmcommand
specifies the command you want to execute. For more information on the commands you can specify with #WM, see Usage Note "2" on page 944.

#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 a not valid 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 these commands with #WM:
   - CP
   - PSCREEN PUT
   - SET RESERVED
   - SET WINDOW
   - WINDOW MAXIMIZE
   - WINDOW MINIMIZE

   Note: The HELP is not a valid command with #WM; however, it *is* a valid command in the WM environment.
3. For an example of the screen you see when you SET FULLSCREEN ON, see *z/VM: CMS User's Guide*.

Messages and Return Codes

- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS914E The maximum number of IUCV connections has been reached [RC=256]
- DMS926E Command is only valid on a display terminal [RC=88]
- DMS927E The physical screen must contain at least 20 lines and 80 columns [RC=24]
- DMS639E Error in NUCXLOAD routine; return code was *nnn* [RC=24]
- DMS931E Invalid WM command: *command*
- DMS1125E *command* is not allowed as an immediate command

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	"Command Syntax Error Messages" on page 1411

SET GEN370

Authorization

General User

Purpose

Use the SET GEN370 command to specify whether a module generated with the 370 option of the GENMOD command should be allowed to execute.

Operands

ON
means modules generated with the GENMOD 370 option will not be executed in an ESA, XA, or XC virtual machine.

OFF
means modules generated with the GENMOD 370 option will be executed in an ESA, XA, or XC virtual machine.

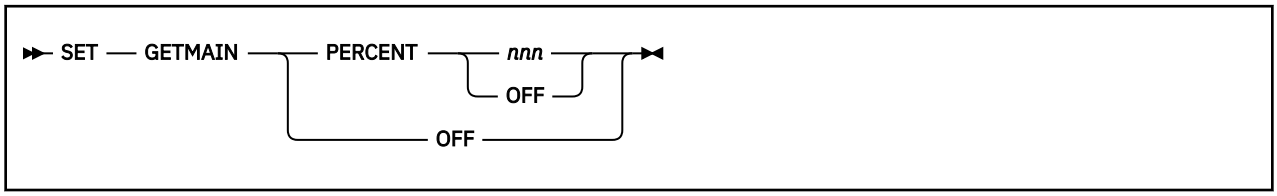
Initial Setting

GEN370 ON

Usage Notes

1. The assumption when the SET GEN370 OFF command is executed is that the CP SET 370ACCOM command is set ON. If the CP SET 370ACCOM command is set OFF, the 370-only modules might fail to execute properly due to dependencies on System/370 instructions.
2. Any modules generated without the 370 option of the GENMOD command are not affected.
3. The GENMOD 370 option is no longer supported.

SET GETMAIN



Authorization

General User

Purpose

Use the SET GETMAIN command to control the amount of storage obtained on a variable GETMAIN request.

Operands

OFF
indicates the maximum amount of storage that can be obtained for a variable GETMAIN request is based on the size of the virtual machine as follows:

VMSIZE	Percentage (approximate)
less than or equal to 4MB	67%
5MB/6MB	80%
greater than 6MB	95%

PERCENT *nnn*
indicates the maximum amount of storage that can be obtained for a variable GETMAIN request is *nnn* percent of the largest contiguous piece of storage available. The variable *nnn* is a decimal number from 1 to 100.

PERCENT OFF
indicates the maximum amount of storage that can be obtained for a variable GETMAIN request is based on the virtual machine size default value.

Initial Setting

GETMAIN OFF

Usage Notes

1. If an OS-based application is exceeding the virtual storage capacity, you can try tailoring the environment with the SET GETMAIN command. If the application is running out of OS (GETMAIN) storage, try increasing the SET GETMAIN PERCENT value. If the application is running out of native CMS storage (CMSSTOR or DMSFREE), try decreasing the SET GETMAIN PERCENT value.
2. Variable GETMAIN requests are those OS storage requests issued with the GETMAIN VC, VU, VRC, and VRU macro forms.

Messages and Return Codes

Return codes:

RC
Meaning

0
Normal completion

24
A not valid parameter was specified for the SET GETMAIN command

System messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

SET IMESCAPE

Authorization

General User

Purpose

Use the SET IMESCAPE command to indicate whether an escape character is required to execute immediate commands.

Operands

ON
indicates an escape character is required to execute immediate commands. The default escape character is a semi-colon (;).

OFF
indicates an escape character is not required to execute immediate commands. This is the default setting.

char
indicates 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

- DMS109S Virtual storage capacity exceeded [RC=104]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

SET IMPCP

Authorization

General User

Purpose

Use the SET IMPCP command to control handling of command names CMS does not recognize. Unknown commands may be considered CP commands or an error.

Operands

ON
passes command names 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

- DMS109S Virtual storage capacity exceeded [RC=104]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

SET IMPEX

Authorization

General User

Purpose

Use the SET IMPEX command to control whether the EXEC files are to be treated as commands.

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, and so forth.

Initial Setting

IMPEX ON

Messages and Return Codes

- DMS109S Virtual storage capacity exceeded [RC=104]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

SET INPUT

Authorization

General User

Purpose

Use the SET INPUT command 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.

Operands

INPUT *a xx*
translates the specified character 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.

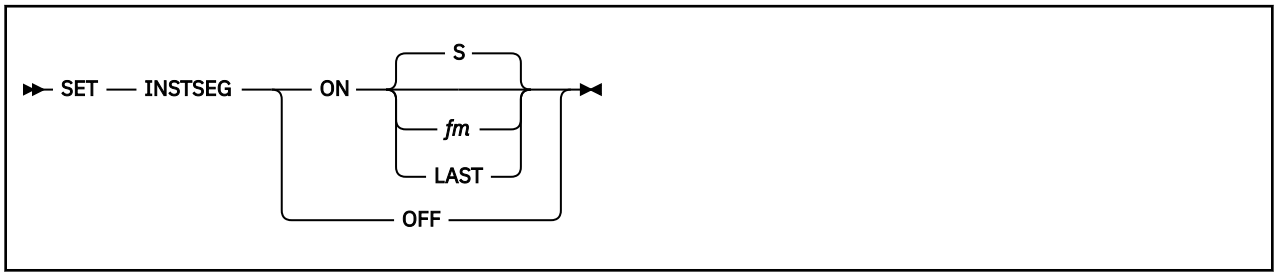
Messages and Return Codes

- DMS109S Virtual storage capacity exceeded [RC=104]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

SET INSTSEG



Authorization

General User

Purpose

Use the SET INSTSEG command to specify whether the system should search the CMS installation saved segment to locate an exec or XEDIT 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.

Operands

ON
indicates 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 you specify. If you have not 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 the saved segment will be searched after all accessed disks and directories have been searched.

OFF
indicates 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

- DMS048E Invalid filemode *mode* [RC=24]
- DMS109S Virtual storage capacity exceeded [RC=104]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

SET KEYPROTECT

Authorization

General User

Purpose

Use the SET KEYPROTECT command to reset the user keys, X'E0', when a DMSFRET or CMSSTOR RELEASE occurs.

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 the user keys are set properly.
2. To check the setting of KEYPROTECT, issue:

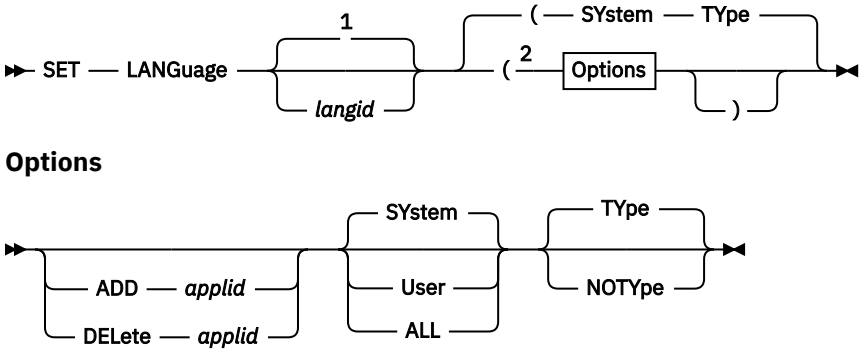
```
QUERY KEYPROTECT
```

Messages and Return Codes

System messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

SET LANGUAGE



Notes:

[1] The default is the current language.
[2] Options entered between the left '(' and right ')' delimiters can be in any order.

Authorization

General User

Purpose

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, for example 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.

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*, SET LANGUAGE gets the one or two-character country code from the VMFNLS LANGLIST for the current language.

Options

ADD
activates the system and user language files, for the application named (*applid*). For more information on making additions to system message repositories and the CMS command syntax file, see *z/VM: CMS Application Development Guide*.

If you specify ADD and a language ID different from the current language setting:

- The current language is changed for all active applications.
- The application named is activated.
- A request is made for CP to change the language used to issue CP messages to your virtual machine.

DELete
deactivates the system and user language files, for the application named.

applid
specifies the application whose system and user language files that should be added or deleted. The application ID must be three characters in length.

User
specifies user repositories, and/or loading user command syntax tables, and/or command synonym tables are loaded into storage or removed from storage. For more information, see Usage Note "1" on page 955.

SYstem
specifies 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 the saved segment with system-provided language files *and* user additions are activated or deactivated.

TYpe
specifies all messages from the SET LANGUAGE command are to be typed to the console. This is the default.

NOTYpe
specifies 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 TXT*langid*, 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

   ***applid*UME*cc* TEXT**
   User message repositories.

   ***applid*UPA*cc* TEXT**
   User command syntax definitions.

   ***applid*USY*cc* 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 *z/VM: CMS Application Development Guide*.

   Where:

   applid
   identifies the three-character application identifier.

   cc
   specifies 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 you can set in CMS.

   Note: 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 z/VM 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, multiple language configurations need to be available for the hardware. This will insure 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 German. 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. 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, place them in shared storage.
8. CMS uses the language built into the CMS nucleus even when a saved segment with system-provided language files for the same language exists.

Examples

If your virtual machine was set to American English (*langid*=AMENG) and you are working in CMS (*applid*=DMS).

- To work in uppercase English (*langid*=UCENG) and run an application called APPLICATION1 (*applid*=AP1), enter:

  ```
  set language uceng ( add ap1 system
  ```

  which changes the language to uppercase English for CMS and APPLICATION1, and informs CP to change its language to uppercase English.
- 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 ( delete ap1 all
  ```

  which returns your virtual machine to where you started: running CMS in American English with no other active applications.

Messages and Return Codes

- DMS005E No application id specified [RC=24]
- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS276E Invalid language id *langid* [RC=24]
- DMS274E The requested language *langid* is not available; *langid* forced [by CP] [RC=104]
- DMS278E Unable to set requested language *langid* [RC=104]

- DMS278E Unable to set requested language *langid*; *langid* forced [RC=104]
- DMS278E Unable to set requested language *langid*; *langid* forced by CP, condition code *cc*, return code *rc* [RC=104]
- DMS279E Application *applid* not found in the language saved segment [RC=28]
- DMS279I Application *applid* not found in the language saved segment
- DMS280E Application *applid* not active [RC=28]
- DMS281E Application DMS cannot be deleted [RC=24]
- DMS283E The *langid* saved segment could not be found; return code *rc* from SEGMENT [RC=128]
- DMS283E The *langid* saved segment could not be loaded; return code *rc* from SEGMENT [RC=128]
- DMS332E No user additions were loaded [RC=28]
- DMS332I No user additions were loaded
- DMS334E No system information or user additions were found for application *applid* [RC=28]
- DMS346E Error *nn* loading *fn ft fm* from disk or directory [RC=32]
- DMS639E Error in *name* routine; return code was *nn*
- DMS770E Invalid application id *applid* [RC=24]
- DMS1229E *fileid* is empty [RC=88]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411
Errors in the Shared File System	“File Pool Server Messages” on page 1414

SET LDRTBLS

Authorization

General User

Purpose

Use the SET LDRTBLS command to define the initial number of pages of storage to be used for loader tables.

Operands

LDRTBLS *nnn*
defines the initial number (*nnn*) 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 storage has 3 pages of loader tables; a larger virtual machine has 4 pages. Each loader table page has a capacity of 169 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 *nnn* command provided:

- *nnn* is a decimal number between 1 and 127 inclusive.
- The virtual machine has enough contiguous storage available to allow *nnn* pages to be used for loader tables.

If these two conditions are met, *nnn* pages are set aside for loader tables.

When the SET LDRTBLS command is used, the existing loader tables are released and storage for the requested number of pages is obtained. Storage will always be requested at X'10000' first. If X'10000' is not available, or the size of the request causes the loader tables to cross X'20000', storage will be obtained from any available location less than 16MB.

Initial Setting

Dependent on size of virtual machine.

Usage Notes

The SET LDRTBLS command may be used during program build (LOAD and INCLUDE command processing), however, it is not needed during program execution. The size of the loader tables will be increased automatically, if necessary, when MODULE files created with the MAP option of the GENMOD command are executed.

Messages and Return Codes

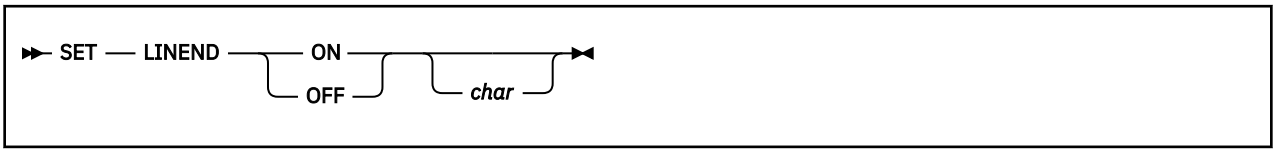
- DMS031E Loader tables cannot be modified [RC=40]
- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS990I Insufficient storage available to create the requested loader tables. The loader tables that existed when the SET LDRTBLS command was issued have been created. [RC=0]

- DMS991E Insufficient storage available to create the loader tables [RC=104]
- DMS992I 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 their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

SET LINEND



Authorization

General User

Purpose

Use the SET LINEND command to activate and define the logical line end for full-screen CMS.

Operands

ON
allows you to enter several commands on the same line, separated by the line end character.

OFF
specifies the logical line end character is not recognized.

char
is the character to be used as a line end character. If *char* is not specified, will remain as the current setting.

Initial Setting

LINEND ON #

Usage Notes

If you redefine the line end character to a symbol other than a pound sign (#), the #WM command is not valid. 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

- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS639E Error in NUCXLOAD routine; return code was *nnn* [RC=24]
- DMS926E Command is only valid in CMS FULLSCREEN mode [RC=88]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

SET LOADAREA

Authorization

General User

Purpose

Use the SET LOADAREA command 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.

Operands

20000
indicates the LOAD command will start loading at X'20000' if ORIGIN, RMODE, or AMODE option is not specified on the LOAD command. If you specified:

- ORIGIN, the load begins at the specified address.
- RMODE, the load begins at the largest contiguous area below 16MB.
- AMODE, the load address is determined from the resulting default RMODE setting.

The X'20000' setting overrides the RMODE definition encountered during text file ESD processing when the ORIGIN, RMODE, or AMODE option is not specified on the LOAD command.

RESPECT
indicates that the LOAD command will respect the RMODE during the text file ESD processing to determine where the loaded programs should reside when the RMODE, ORIGIN, or AMODE option is not specified on the LOAD command.

If you specified RMODE 24 or RMODE ANY on the first text file ESD, loading will begin at the largest contiguous area of storage allocated:

- Below 16MB for RMODE 24
- Above 16MB for RMODE ANY

Initial Setting

RESPECT

Examples

These tables show how the setting of the SET LOADAREA command affects various kinds of program loads done with the LOAD command.

1. SET LOADAREA 20000 active

Table 44. Effects of Specifying SET LOADAREA 20000 on LOAD Commands

LOAD command	Effect
LOAD pgma ...	Load begins at 20000, overrides RMODE on text(s) ESD, AMODE on text file ESD respected

Table 44. Effects of Specifying SET LOADAREA 20000 on LOAD Commands (continued)

LOAD command	Effect
LOAD pgma ..(RMODE 24	CMS loads pgma at largest contiguous free storage area under 16MB.
LOAD pgma ..(RMODE ANY	CMS loads pgma at largest contiguous free storage area under 16MB.
LOAD pgma ..(ORIGIN TR	Load begins at start of transient area, SET LOADAREA is overridden, AMODE on text file ESD respected unless AMODE override on LOAD command
LOAD pgma ..(ORIGIN *hexloc*	Load begins at area specified by hexloc, SET LOADAREA is overridden, AMODE on text file ESD respected unless AMODE override on LOAD command

2. SET LOADAREA RESPECT active

Table 45. Effects of Specifying SET LOADAREA RESPECT on LOAD Commands

LOAD command	Effect
LOAD pgma ...	Load begins at the largest contiguous area according to RMODE determined from text(s) ESD, AMODE on text file ESD respected
LOAD pgma ..(RMODE 24	Load begins at the largest contiguous area under 16MB, AMODE on text file ESD respected unless AMODE override on LOAD command
LOAD pgma ..(RMODE ANY	Load begins at the largest contiguous area above 16MB, AMODE on text file ESD respected unless AMODE override on LOAD command
LOAD pgma ..(AMODE 24, 31 or ANY	Load begins at the area determined from the default RMODE setting. For more information, see Table 20 on page 476
LOAD pgma ..(ORIGIN TR	Load begins at start of transient area, SET LOADAREA is overridden, AMODE on text file ESD respected unless AMODE override on LOAD command
LOAD pgma ..(ORIGIN *hexloc*	Load begins at area specified by hexloc, SET LOADAREA is overridden, AMODE on text file ESD respected unless AMODE override on LOAD command

Messages and Return Codes

System messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

SET LOCATION

Authorization

General User

Purpose

Use the SET LOCATION command to display the location indicator in the window when the data in the virtual screen exceeds the size of the window.

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. (However, if the border is displayed using programmed symbol set 1 (PS1), the default programmed symbol set is used (PS0).) For example:

```
Lines 25 - 44 of 44
Columns 1 - 20 of 80
```

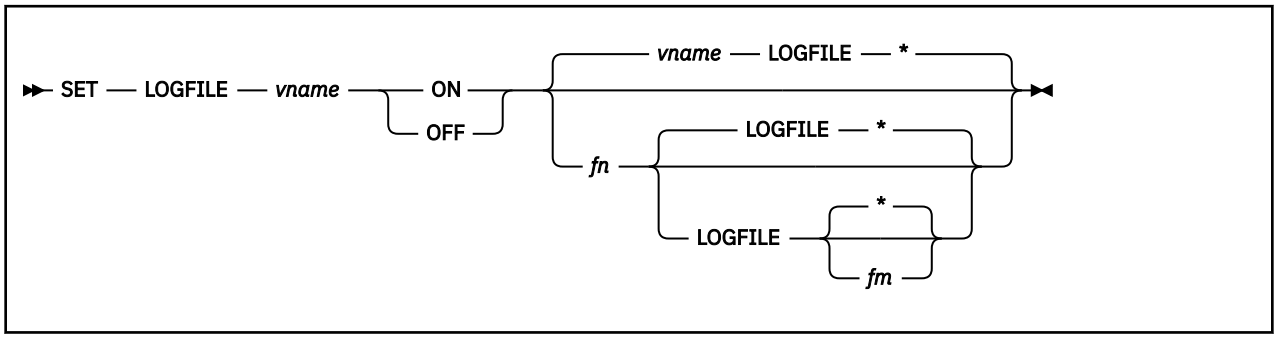
Messages and Return Codes

- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS639E Error in NUCXLOAD|NUCXDROP routine; return code was *nnn* [RC=24]
- DMS921E Window *wname* is not defined [RC=28]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

SET LOGFILE



Authorization

General User

Purpose

Use the SET LOGFILE command to control whether a log file is updated with data written to a virtual screen.

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. For more information, see "Usage Notes" on page 965.

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 or a file mode, 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:ss
```

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.
7. Logging occurs when the output text is moved from the queue to the virtual screen data buffer. This occurs when:
   - VSCREEN WAITREAD or VSCREEN WAITT has been issued for the particular virtual screen that has output data queued.
   - PSCREEN REFRESH has been issued.
   - A limit has been reached for the number of lines queued.

   For example, consider this scenario:

   a. Logging is in effect when a program writes three lines to a virtual screen.
   b. However, logging is set off for that particular virtual screen before the program exits.
   c. Therefore, you may not see the data logged in the log file. If the program had issued VSCREEN WAITREAD, VSCREEN WAITT, or PSCREEN REFRESH before exiting, then the data would have been moved from the queue to the virtual screen data buffer.

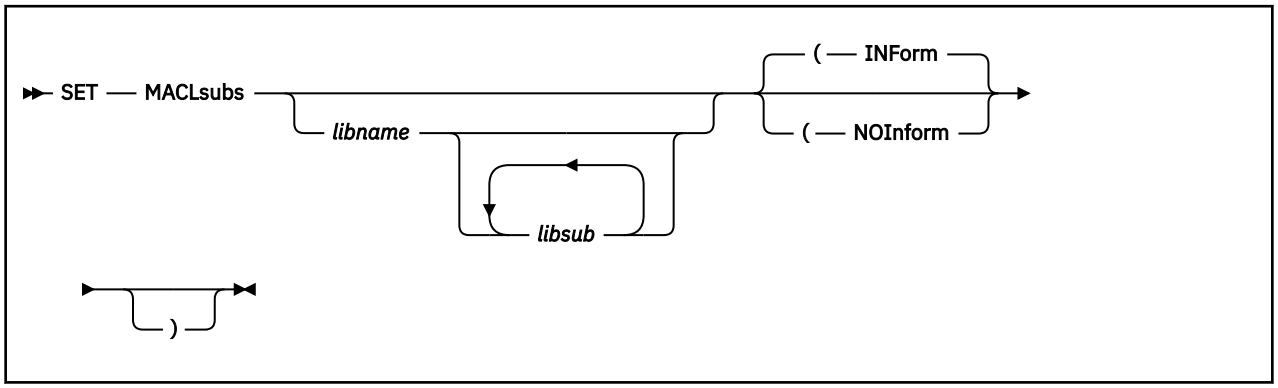
Messages and Return Codes

- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS639E Error in NUCXLOAD routine; return code was *nnn* [RC=24]
- DMS921E Virtual screen *vname* is not defined [RC=28]
- DMS928E 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 their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

SET MACLSUBS



Authorization

General User

Purpose

Use the SET MACLSUBS command to set a macro library substitution that takes effect when the GLOBAL command is entered. The SET MACLSUBS command is the first step in the library substitution process. The second step is the GLOBAL command that does the actual replacement defined by the SET MACLSUBS command.

Operands

libname
is the file name of the macro library with a file type of MACLIB.

libsub
is one or more file names of the macro libraries with a file type of MACLIB to be replaced by the GLOBAL command.

Options

INForm
specifies the informational message (DMS2152I) is issued when the GLOBAL command is replacing macro libraries. This is the default.

NOInform
specifies the informational message (DMS2152I) is not issued when the GLOBAL command is replacing macro libraries.

Usage Notes

1. Enter the SET MACLSUBS command before the GLOBAL command for the substitutions to take effect.
2. The library substitutions are identified by the GLOBAL command in the order in which they are specified by the SET MACLSUBS command.
3. The SET MACLSUBS command remains in effect for an entire CMS session unless it is explicitly canceled or re-entered. If a program failure forces you to IPL CMS again, you must re-enter the SET MACLSUBS command.
4. There are no default libraries. If you choose to substitute the VM libraries during a session, enter the MACLMIG command.
5. To find out what libraries have substitutions, use the QUERY MACLSUBS command.

6. To clear the library substitutions or change INFORM/NOINFORM for a specific library, enter the command without the *libsub* operand. There are several variations:
   - `SET MACLSUBS`

     Clears all the library substitution settings.
   - `SET MACLSUBS (INFORM`

     Changes all the library substitution settings to display the informational message when GLOBAL is replacing the library names.
   - `SET MACLSUBS` *`libname`* `(NOINFORM`

     Changes the *libname* status and the GLOBAL command does not display the informational message for *libname*.
   - `SET MACLSUBS` *`libname`*

     Clears the library substitution setting of *libname*.
7. There is no limit to the number of library substitutions you can specify. However, the list of library substitutions is subject to other system limits such as command line length and the GLOBAL command limitations.
8. This command replaces any previous substitutions for the specified library name.
9. You may receive the DMS108S message from the GLOBAL command even though you specify less than the maximum of 63 library names. This message results because a library name may be replaced by more than one substitute library name.
10. Macro library substitutions are not substituted. For example, if you enter the following two commands:

    ```
    SET MACLSUBS LIBA LIBB
    SET MACLSUBS LIBB LIBC
    ```

    the GLOBAL command will not substitute LIBC for LIBA. When you specify `GLOBAL MACLIB LIBA`, the result is LIBB (not LIBC). When you specify `GLOBAL MACLIB LIBB`, the result is LIBC.

Examples

If you want to replace the LIBR1 library with the MACL1 and MACL2 macro library names and be notified when GLOBAL is replacing the macro libraries, enter:

```
set maclsubs libr1 macl1 macl2 (inform
```

Then, when you enter:

```
global maclib libr1
```

the message is:

```
DMS2152I  LIBR1 specified, library substitution name(s) used
```

Messages and Return Codes

- DMS2152I *libname* specified, library substitution name(s) used

SET NONDISP

Authorization

General User

Purpose

Use the SET NONDISP command to define a character for CMS and XEDIT that is displayed in place of nondisplayable characters.

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” on page 954.
2. Changing the NONDISP setting for CMS also changes the NONDISP setting for XEDIT, as well changes to the NONDISP setting for XEDIT will change the NONDISP setting for CMS
3. Changing the NONDISP character does not change characters already displayed on the screen unless that line is altered.
4. If a double-byte (DBCS) equivalent exists for the single-byte character you specify, it will be assigned at the same time as the single-byte character. The double-byte nondisplayable character will be the equivalent character from the Ward 42 code page. You cannot directly enter double-byte nondisplayable characters.

   If the single-byte character you entered is not valid (cannot be displayed on your terminal), neither the single-byte nor the double-byte nondisplayable character is changed. If the single-byte character you entered is valid but there is no double-byte equivalent, the single-byte nondisplayable character is changed, but the double-byte nondisplayable character remains at the initial setting.

   The QUERY NONDISP command and the XEDIT EXTRACT and QUERY subcommands will return only the single-byte nondisplayable character, even if the double-byte nondisplayable character is different.

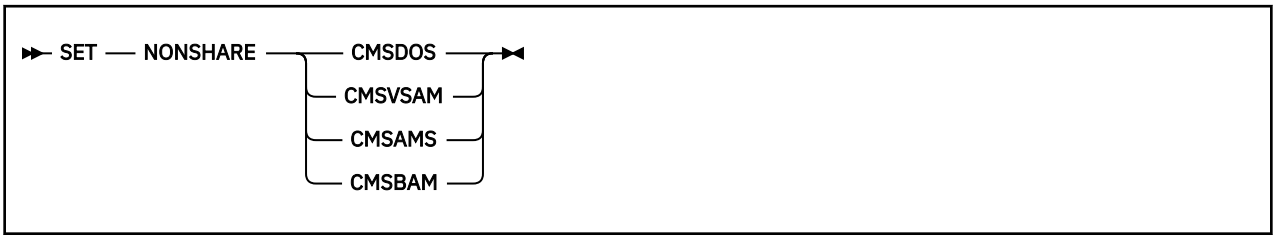
Messages and Return Codes

- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS639E Error in NUCXLOAD routine; return code was *nnn* [RC=24]
- DMS926E Command is only valid on display terminal [RC=88]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

SET NONSHARE



Authorization

General User

Purpose

Use the SET NONSHARE command to get your own copy of a shared named system.

Operands

CMSDOS
CMSVSAM
CMSAMS
CMSBAM
specifies the shared named system for which you want a nonshared copy.

Messages and Return Codes

- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS400S System *sysname* does not exist [RC=44]
- DMS401S VM size (*size*) cannot exceed *sysname* start address (*vstor*) [RC=104]
- DMS410S Control program error indication *xxx* [RC=*rc*]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

Note: In RC=*rc*, the *rc* represents the actual error code generated by CP.

SET OLDCMDS

Authorization

General User

Purpose

Use the SET OLDCMDS command to determine whether the original formats of the Session Services commands are available for invocation. The original formats are no longer available. The OLDCMDS setting will remain OFF, regardless of the operand you specify. You must use the current formats.

This command is maintained for compatibility purposes only.

Original Format	Current Format
CONVERT COMMANDS	GENCMD
PUT SCREEN	PSCREEN PUT
REFRESH	PSCREEN REFRESH
ALARM VSCREEN	VSCREEN ALARM
CLEAR VSCREEN	VSCREEN CLEAR
CURSOR VSCREEN	VSCREEN CURSOR
DEFINE VSCREEN	VSCREEN DEFINE
DELETE VSCREEN	VSCREEN DELETE
GET VSCREEN	VSCREEN GET
PUT VSCREEN	VSCREEN PUT
ROUTE	VSCREEN ROUTE
WAITREAD VSCREEN	VSCREEN WAITREAD
WAITT VSCREEN	VSCREEN WAITT
WRITE VSCREEN	VSCREEN WRITE
SCROLL	WINDOW BACKWARD WINDOW BOTTOM WINDOW DOWN WINDOW FORWARD WINDOW LEFT WINDOW NEXT WINDOW RIGHT WINDOW TOP WINDOW UP
CLEAR WINDOW	WINDOW CLEAR
DEFINE WINDOW	WINDOW DEFINE
DELETE WINDOW	WINDOW DELETE
DROP WINDOW	WINDOW DROP
HIDE WINDOW	WINDOW HIDE

Original Format	Current Format
MAXIMIZE WINDOW	WINDOW MAXIMIZE
MINIMIZE WINDOW	WINDOW MINIMIZE
POP WINDOW	WINDOW POP
POSITION WINDOW	WINDOW POSITION
RESTORE WINDOW	WINDOW RESTORE
SHOW WINDOW	WINDOW SHOW
SIZE WINDOW	WINDOW SIZE

SET OSTXTBUF

Authorization

General User

Purpose

Use the SET OSTXTBUF command to control the size of the gap in storage reserved before OS loading a text file.

Operands

hexnum
indicates *hexnum* bytes of storage will be added to the current location where an OS loaded text file will be loaded into virtual storage, where *hexnum* is a hexadecimal number between X'50000' and X'1000000'.

Note: The *hexnum* will always be rounded to a doubleword boundary.

Initial Setting

OSTXTBUF 50000

Usage Notes

If an OS loaded application is being overlaid by a subsequent load of a nonrelocatable module, try using the SET OSTXTBUF command to increase the OSTXTBUF setting to a size greater than the cumulative size of the nonrelocatable programs.

The OSTXTBUF size will not be used for programs executing above the 16MB line in an XA or XC mode virtual machine. These applications must create relocatable modules to avoid the overlay problem noted above.

Messages and Return Codes

- DMS026E Invalid parameter *hexnum* for OSTXTBUF parameter [RC=24]

Return codes:

RC
Meaning

0
Normal completion

24
A not valid parameter was specified for the SET OSTXTBUF command

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

SET OUTPUT

Authorization

General User

Purpose

Use the SET OUTPUT command to translate a hexadecimal representation displayed at the terminal to a specified character.

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 at the end of a line.
4. You can use the ADDRESS COMMAND environment with SET OUTPUT from a REXX exec to cause translation to lowercase characters. Following is a sample exec which translates uppercase A's to lowercase a's:

```
/*                                          */
/* Translate uppercase A (X'C1') to lowercase */
/*                                          */
address command 'SET OUTPUT C1 a'
exit
```

Messages and Return Codes

- DMS109S Virtual storage capacity exceeded [RC=104]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

SET PROTECT

Authorization

General User

Purpose

Use the SET PROTECT command to specify whether the CMS nucleus is protected against writing in its storage area.

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

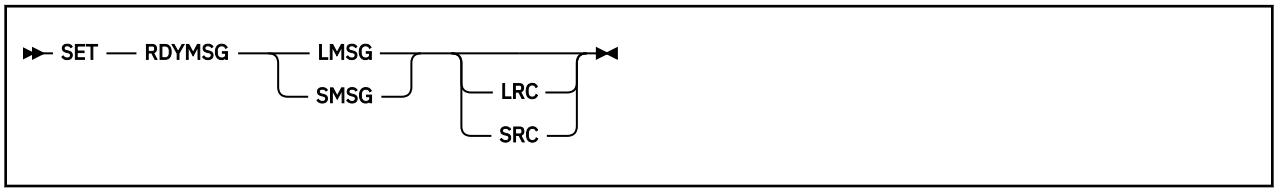
Messages and Return Codes

- DMS109S Virtual storage capacity exceeded [RC=104]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

SET RDYMSG



Authorization

General User

Purpose

Use the SET RDYMSG command to specify whether CMS issues the standard CMS ready message or a shortened form, and the standard five digit return code or an expanded ten digit return code.

Operands

LMSG
indicates 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 a shortened form of the CMS ready message (Ready;) which does not include the time is used.

LRC
Indicates an expanded return code (up to ten digits, plus a minus sign) is issued with the ready message when an error occurs. If the return code is not greater than five digits, the standard five digit return code is issued with leading zeros.

For example, with the LRC operand in effect, the return code displays the following ready message:

RC=24:
Ready; (00024)

RC=-24:
Ready; (-0024)

RC=200000:
Ready; (200000)

RC=-200000:
Ready; (-200000)

SRC
Indicates the standard five digit return code is issued with the ready message when an error occurs. If the return code is greater than five digits, truncation occurs.

For example, with the SRC operand in effect, the return code displays the following ready message:

RC=24:
Ready; (00024)

RC=-24:
Ready; (-0024)

RC=200000:
Ready; (00000)

RC=-200000:
Ready; (-0000)

Initial Setting

RDYMSG LMSG LRC

Messages and Return Codes

- DMS109S Virtual storage capacity exceeded [RC=104]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

SET RECALL

Authorization

General User

Purpose

Use the SET RECALL command to specify whether the SFS files that have been placed in migrated status by DFSMS/VM (that is, have had their data moved into the DFSMS/VM storage repository) are to be automatically recalled when file data is referenced.

Operands

ON
indicates you want files in DFSMS/VM migrated status (that is, files whose data has been moved into the DFSMS/VM storage repository) to be implicitly recalled when referred to and required as input.

OFF
indicates referencing data in files in DFSMS/VM migrated status will not cause the data to be implicitly recalled. When RECALL is OFF, you must issue the DFSMS RECALL command to recall such a file prior to opening it for input or update.

Initial Setting

RECALL ON

Usage Notes

The RECALL setting is the same for all your file pools.

For more information on how to determine your current RECALL setting, see "QUERY RECALL" on page 745. For more information about DFSMS/VM commands, see *z/VM: DFSMS/VM Storage Administration*.

Note: The RECALL setting in your virtual machine has no effect when you are replacing the contents of a migrated file. For example, if SET RECALL is OFF and you issue:

```
copyfile fromfn ft fm  tofn ft fm (replace
```

and the file `tofn` was migrated before the command was issued, COPYFILE will complete successfully. At the completion of the command, `tofn` will no longer be migrated.

If SET RECALL is ON and both `fromfn` and `tofn` were migrated, `fromfn` would be recalled, and `tofn` would be replaced with the `fromfn` data. At the completion of this COPYFILE, neither `fromfn` nor `tofn` would be migrated.

If you issue this same command with SET RECALL OFF, the command would fail and the files would remain in migrated status.

Messages and Return Codes

- DMS109S Virtual storage capacity exceeded [RC=104]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

SET REDTYPE

Authorization

General User

Purpose

Use the SET REDTYPE command to have CMS error messages typed in red for certain terminals equipped with the appropriate terminal feature and a two-color ribbon.

Operands

ON
types CMS error messages in red for certain terminals equipped with the appropriate terminal feature and a two-color ribbon.

OFF
suppresses red typing of error messages.

Initial Setting

REDTYPE OFF

Messages and Return Codes

- DMS109S Virtual storage capacity exceeded [RC=104]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

SET RELPAGE

Authorization

General User

Purpose

Use the SET RELPAGE command to release page frames of storage and set them to binary zeros, or to hold the pages of storage.

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 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

- DMS109S Virtual storage capacity exceeded [RC=104]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

SET REMOTE

Authorization

General User

Purpose

Use the SET REMOTE command to control the compression of data sent to the terminal for CMS and XEDIT.

Operands

ON
specifies 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 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.

Note: If a terminal is connected through VTAM or PVM, or is defined as a logical device, it will appear to CMS as a local terminal.

Usage Notes

Changing the REMOTE setting for CMS also changes the REMOTE setting for XEDIT, and changing the REMOTE setting for XEDIT also changes the REMOTE setting for CMS.

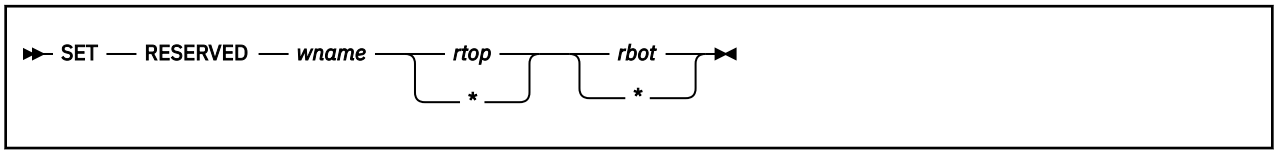
Messages and Return Codes

- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS639E Error in NUCXLOAD routine; return code was *nnn* [RC=24]
- DMS926E 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 their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

SET RESERVED



Authorization

General User

Purpose

Use the SET RESERVED command to specify the maximum number of lines in a window that displays the virtual screen reserved lines.

Operands

wname
is the name of the window.

rtop
is the maximum number of reserved lines 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.

rbot
is the maximum number of reserved lines 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.

For more information, see Usage Notes "2" on page 985 and "3" on page 986.

Initial Setting

RESERVED *wname* * *

Usage Notes

1. Reserved lines are maintained in the virtual screen buffers and are often 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. For more information regarding virtual screen reserved lines, see "VSCREEN DEFINE" on page 1174 and "VSCREEN WRITE" on page 1192.
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 it is now only 3 lines long, the window is displayed with 1 top reserved line and 2 bottom reserved lines, according to these rules and Usage Note "3" on page 986.

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

- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS639E Error in NUCXLOAD routine; return code was *nnn* [RC=24]
- DMS921E Window *wname* is not defined [RC=28]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	"Command Syntax Error Messages" on page 1411

SET RORESPECT

Authorization

General User

Purpose

Use the SET RORESPECT command to specify whether the XEDIT and COPYFILE commands should respect the read-only access mode of SFS file control directories.

Operands

ON
indicates you want XEDIT and COPYFILE to respect a read-only access mode of an SFS file control directory. When an update attempt is made on a R/O directory and SET RORESPECT is ON, the update will fail.

OFF
indicates attempts to update files through the XEDIT and COPYFILE commands in the SFS file control directories will be based on authorization regardless of the access mode.

Initial Setting

RORESPECT OFF

Usage Notes

The RORESPECT setting applies to the file mode specified in the XEDIT or COPYFILE command. For example:

```
SET RORESPECT ON
ACCESS .MYDIR B (FORCERW
```

which contains file 'BASE FILE'

```
ACCESS .MYDIRALIAS C (FORCERO
```

which contains 'ALIAS FILE' which is an alias to 'BASE FILE'.

```
COPYFILE SOME FILE A ALIAS FILE C
```

fails because file mode c is read-only and RORESPECT is ON.

Compare this with:

```
SET RORESPECT ON
ACCESS .MYDIR B (FORCERO
```

which contains file 'BASE FILE'

```
ACCESS .MYDIRALIAS C (FORCERW
```

which contains 'ALIAS FILE' which is an alias to 'BASE FILE'.

```
COPYFILE SOME FILE A ALIAS FILE C
```

works because file mode c is read-write (and the file is updated).

Messages and Return Codes

- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS389E Invalid operand: *operand* [RC=24]
- DMS391E Unexpected operand(s): *operand* [RC=24]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

SET SERVER

Authorization

General User

Purpose

Use the SET SERVER command to enable private resource processing.

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 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. For more information on private resource processing, see *z/VM: CMS Application Development Guide* and *z/VM: Connectivity*.

Messages and Return Codes

- DMS1256E 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 their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

SET STORECLR

Authorization

General User

Purpose

Use the SET STORECLR command to set the point of automatic GETMAIN storage cleanup and determine the action for user invocation of the CMS STRINIT macroinstruction.

Operands

ENDCMD
indicates CMS returns GETMAIN storage at end-of-command (the point where the CMS ready message (Ready;) is displayed) and CMS honors user invocations of STRINIT and EXECOS.

ENDSVC
indicates CMS returns GETMAIN storage at SVC 202, SVC 204/CMSCALL, and SVC 42/ATTACH termination. ENDSVC also indicates CMS treats user invocations of STRINIT and EXECOS 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 affect the method of GETMAIN storage cleanup at ABEND recovery. All GETMAIN storage is reclaimed at ABEND.
4. If an entire system must retain GETMAIN storage, place the SET STORECLR ENDCMD command in the SYSTEM profile. If a particular virtual machine must retain GETMAIN storage, place the SET STORECLR ENDCMD command in the user's PROFILE EXEC. For a particular application, you can switch STORECLR from ENDCMD to ENDSVC and back using the CMSCALL macro for assembler programs, or by using a front-end EXEC.

   Note: When you use the SET STORECLR command to change the current setting, a STRINIT is performed and existing OS/MVS GETMAIN storage is cleaned up.

Messages and Return Codes

Return codes:

RC
Meaning

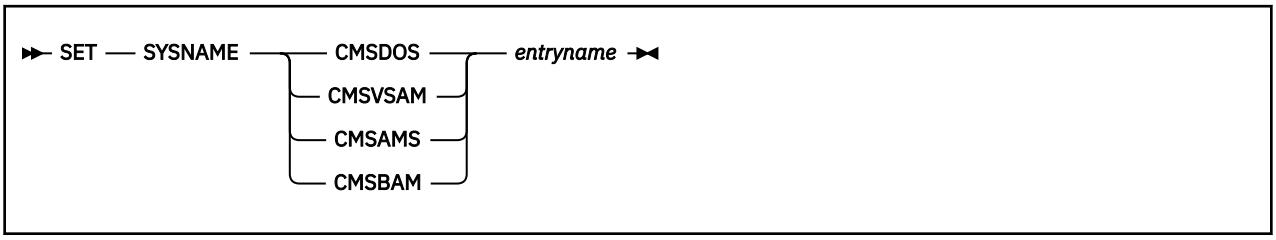
0
Normal completion.

24
A not valid parameter was specified on the SET STORECLR command.

System messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax.	“Command Syntax Error Messages” on page 1411

SET SYSNAME



Authorization

General User

Purpose

Use the SET SYSNAME command to replace a saved system name entry in the SYSNAMES table with the name of an alternate, or backup system.

Operands

CMSDOS
CMSVSAM
CMSSAMS
CMSBAM
allows you to replace a saved system name entry in the SYSNAMES table with the name of an alternative, or backup system.

Usage Notes

A separate SET SYSNAME command must be issued for each saved system 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

- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS142S Saved system name *sysname* invalid [RC=24]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

SET TAPECSL

Authorization

General User

Purpose

Use the SET TAPECSL command to change the setting of CMS OS Simulation tape action for calling DFSMS/RMS via CSL for tape mounts or demounts on an automated tape librarian controlled tape drive.

Operands

ON
the CMS OS Simulation routines will use CSL to call DFSMS/RMS functions to either mount or demount tapes on an automated tape librarian controlled tape drive. This is the default.

OFF
the CMS OS Simulation routines will not use CSL to call DFSMS/RMS functions to either mount or demount an automated tape librarian controlled tape drive. Instead, native CMS OS Simulation routines will issue messages to an operator to mount a tape and use the TAPE RUN command to unload a tape from the drive.

Initial Setting

TAPECSL ON

Usage Notes

1. The TAPECSL setting determines whether CMS OS Simulation native tape handling routines will directly call DFSMS/RMS functions via the CSL call interface.
2. The TAPECSL setting will not affect vendor product calls to DFSMS/RMS via the CSL interface, or calls to DFSMS/RMS from other IBM products.
3. You can issue SET TAPECSL from a PROFILE EXEC to turn this calling service ON or OFF for a particular CMS user or server virtual machine.
4. You can issue SET TAPECSL from the SYSPROF EXEC (or a local EXEC called from SYSPROF) to run this calling ON or OFF for all CMS users.

Messages and Return Codes

System messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

SET TAPENEVR

Authorization

General User

Purpose

Use the SET TAPENEVR command to control CMS OS Simulation tape label date checking for 'Unexpired Files'.

Operands

ON
specifies existing 'Never Expire' standard tape label Julian dates of ' 99365' and ' 99366' are to be honored as not expired by CMS OS Simulation tape label processing routines and an 'Unexpired File' prompt is to be issued to the user.

OFF
specifies date values in standard tape labels are to be treated as normal dates. Existing 'Never Expire' date values are not honored as special values. The label date is tested against the current system date. If the label date is greater than the current date, CMS OS Simulation causes an 'Unexpired File' prompt. Otherwise, the label date is considered expired, and the tape is writable.

Initial Setting

TAPENEVR ON

Usage Notes

1. The TAPENEVR setting determines whether CMS OS Simulation tape handling routines prompt the user if they find an existing 'Never Expire' expiration date in the tape HDR1 label.
2. Tapes prepared for use with the TAPE WVOL1 command or written as normal standard labeled tape output on 12/31/1999 are given a default Julian expiration date of ' 99365', one of the existing 'Never Expire' values. To prevent unwanted prompting when these tapes are used on or after 12/31/1999, issue the SET TAPENEVR OFF command before processing the tapes.
3. If you use standard labeled tapes for output in either a batch machine or a disconnected server virtual machine, you may want to issue SET TAPENEVR OFF in the PROFILE EXEC or the application REXX EXEC to prevent unwanted message prompting that could adversely affect the processing of the tapes.
4. You can issue SET TAPENEVR OFF from the SYSPROF EXEC (or a local EXEC called from SYSPROF) to turn off 'Never Expire' date prompting for all CMS users.

Messages and Return Codes

System messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

SET TEXT

Authorization

General User

Purpose

Use SET TEXT command to activate character code conversion for TEXT characters for XEDIT and CMS.

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 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 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 XEDIT.

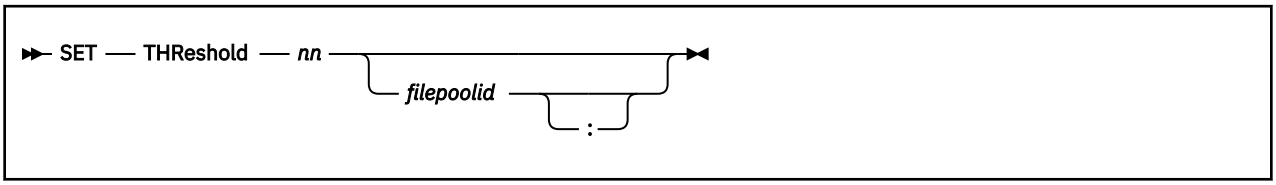
Messages and Return Codes

- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS329W Warning: APL/TEXT option not in effect
- DMS524W NONDISP character reset to ".
- DMS639E Error in NUCXLOAD routine; return code was *nnn* [RC=24]
- DMS926E 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 their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

SET THRESHOLD



Authorization

General User

Purpose

Use the SET THRESHOLD command to indicate when you want a warning message or return code issued to tell you a specified amount of your allocated file space in a file pool has been used.

Note: A different version of this command is available for users with file pool administration authority. For more information, see *z/VM: CMS File Pool Planning, Administration, and Operation*.

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
filepoolid:
specifies in which file pool to set the threshold. If not specified, the default file pool identifier is used. The colon is optional.

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. However, you will only see this message once between console reads. (For example, a console read occurs when you press Enter, so you would only see the message once until you press Enter, even if, for example, you are executing a program which performs multiple operations which exceed the threshold.)
4. If the SET THRESHOLD command is issued from an exec or assembler program for a file pool that is active on the specified work unit, the command will fail.
5. If you are using OS simulation when writing to files in a pre-z/VM release SFS server, the user threshold warning is treated as a disk full error. This could occur on a write or put operation. It could also occur on a read operation because CMS buffers requests.

Examples

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.

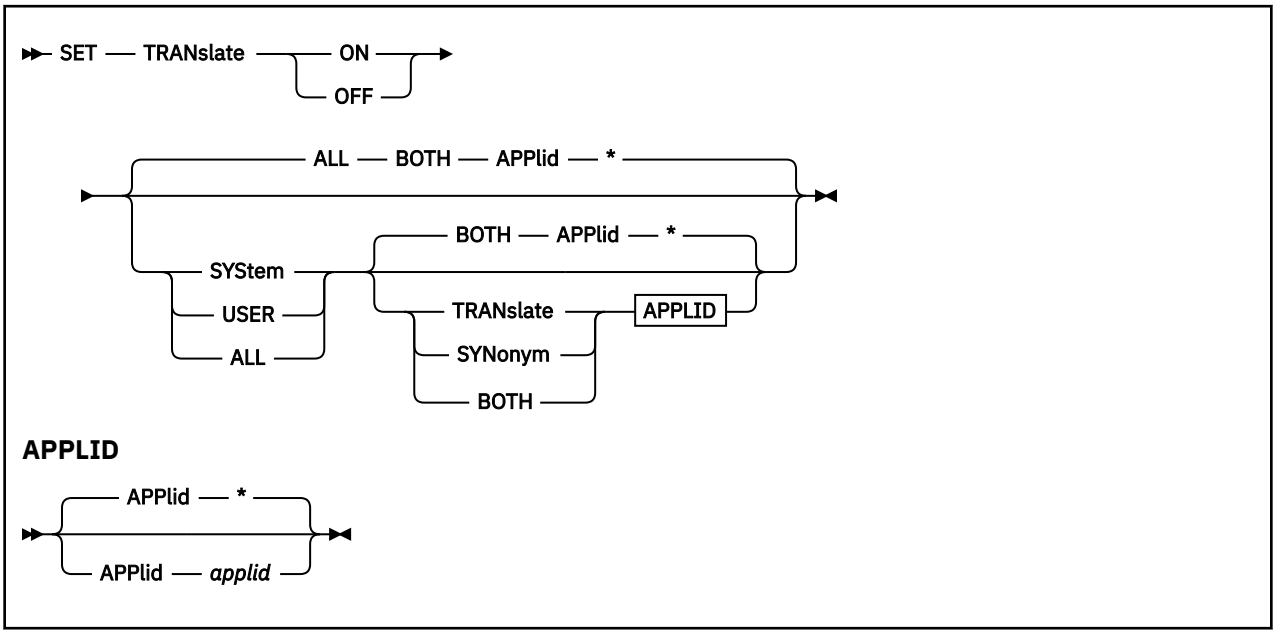
Messages and Return Codes

- DMS1140E You are not enrolled in file pool *filepoolid* [RC=40]
- DMS1141W User filespace threshold still exceeded for the file pool *filepoolid* [RC=4]
- DMS1168E Invalid threshold value *value* [RC=24]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411
Errors in the Shared File System	“File Pool Server Messages” on page 1414

SET TRANSLATE



Authorization

General User

Purpose

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.

Operands

ON
allows you to specify the table to be used for command name translation.

OFF
allows you to specify a table will not be used for command name translation.

ALL
translates command names using both the User and System National Language Translation Tables. This is the default.

BOTH
indicates both national language translations and translation synonyms are set ON or OFF. This is the default.

SYStem
translates command names using only the System National Language Translation Table.

USER
translates command names using only the User National Language Translation Table.

TRANslate
indicates only the national language translations are set ON or OFF.

SYNonym
indicates only the national language 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 a not valid function and the implied CP function is in effect, you may receive message HCPCF003E INVALID OPTIONS - option.
2. If a not valid SET command function is specified from an exec and the implied CP function is in effect, 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.

Examples

To set the translation synonym table OFF for all applications, enter:

```
set translate off user syn
```

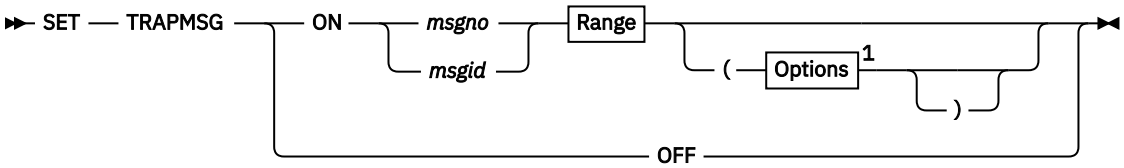
Messages and Return Codes

- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS258E {user|system} translation synonyms can not be set ON unless {user|system} translations are also set ON, application id: *applid* [RC=28]
- DMS258E {user|system} translations can not be set OFF unless {user|system} translation synonyms are also set OFF, application id: *applid* [RC=28]
- DMS280E Application *applid* not active [RC=28]

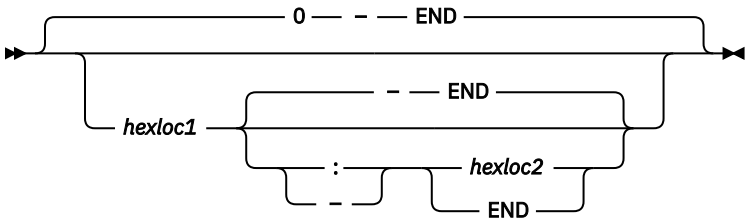
Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

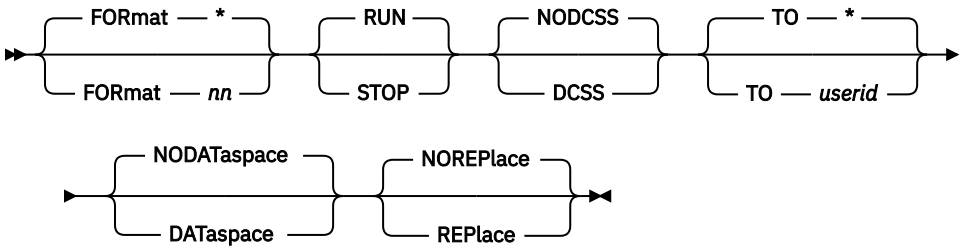
SET TRAPMSG



Range



Options



Notes:

[1] You can enter Options in any order between the parentheses.

Authorization

General User

Purpose

Use the SET TRAPMSG command when a CMS problem is encountered that cannot be explained and it is necessary to capture a sequence of events when a particular message ID occurs. The SET TRAPMSG command is set to indicate which message should cause a trap to spring, and optionally, to specify how much storage to dump.

The message trap can be activated by issuing SET TRAPMSG ON with the respective message ID and dump range. It can be used whenever a message is received that needs further explanation or debugging to find the cause of the message. SET TRAPMSG should generally be used only when contact of a support group for further analysis is necessary. For example, use the SET TRAPMSG command to cause a VMDUMP to be taken when the specified message ID is displayed.

Operands

ON
Turn the message trap on.

OFF
Turn the message trap off.

msgid
The *msgid* operand must be in this *xxxmmm####s* format. Where:

xxx
specifies the prefix or application identifier. For example, DMS, is a three character alphanumeric string therefore, the first character must be alphabetic.

mmm
specifies the module code (a three character alphanumeric string; other valid characters that can indicate which module generated the message are $, #, @, +, –, :, and _). '???' can be used as the wild card or the place holder.

###* or *####
specifies the message number.

s
specifies the severity code. A '?' can be used as a wild card or a place holder.

msgno
The message number is a 1-4 digit decimal number, ranging from 0-9999. If you choose to specify this format, an example of the message header might look like DMS??? *###*? or DMS??? *####*?. '?' means wild card.

hexloc1
0
The *hexloc1* operand is the starting virtual storage address dumped. If the value you specify for *hexloc1* is not a multiple of 4 kilobytes (KB), the control program (CP) will round the *hexloc1* specified down to a 4KB boundary. If you omit the *hexloc1* operand, the default is zero. You may also specify *hexloc2* with the *hexloc1* operand. *hexloc1* has to be in the range of your virtual machine and it cannot be greater than the *hexloc2*.

:
–
are symbols used with *hexloc1* and *hexloc2* to define the virtual storage address range to be dumped.

hexloc2
END
The *hexloc2* operand is the ending virtual storage address dumped. *hexloc2* has to be in the range of your virtual machine and it cannot be less than *hexloc1*. If the value you specify for *hexloc2* is not equal to the starting location (*hexloc1*) and is not a multiple of 4KB, CP will round the *hexloc2* up to the next 4KB boundary. If you do not specify the *hexloc2* operand, the default is END, which is *vmsize-1*. *vmsize* is the defined size of the virtual machine.

FORmat *
FORmat *nn*
Specifies a one or two digit format number. This identifies different versions of the message text that have the same message number. The formats are numbered from 1-99. The default is *, which means any format number. A format of zero is not allowed.

RUN
STOP
The STOP option stops CMS and goes to CP when the message trap springs, but before the dump is generated. This allows you to examine storage, set a trace, or do other CP commands after the message trap springs. Enter a B to take the dump and resume processing. The default is RUN. The STOP option will place the virtual machine into CP READ mode after a message trap springs.

NODCSS
DCSS
Specifies CP takes a dump of all the user discontinuous saved segments not within user storage. If the user has DCSSs that are wholly within user storage, these are not added to the dump in response to the DCSS option. If a DCSS is partly within user storage and partly outside, only those segments outside user storage are dumped in response to this option. The default is NODCSS.

TO *
TO ***userid***
Transfers the dump to the virtual card reader of the specified user ID. If you enter an asterisk (*) after the keyword T0, or do not specify the T0 at all, the system sends the dump to the virtual card reader of the virtual machine you issued the SET TRAPMSG command from. The default is TO *.

NODATaspace
DATaspace
This option dumps SFS data space storage associated with the VM user ID. This option only applies to CMS SFS data space. The default is NODATASPACE.

REPlace
NOREPlace
Specify REPLACE to replace an existing message trap. If REPLACE is not specified and a message trap is already active, an error message will be issued. The default is NOREPLACE.

Usage Notes

1. The default dump range is ‘0 to vmsize-1’ unless specified upon invocation.
2. If only one address is specified for the dump range, it will be set as the starting dump range. The ending dump range will be set to vmsize-1.
3. The message trap will stay active even after a trap is sprung and dumps have been taken. Issue SET TRAPMSG OFF to turn the message trap off.
4. **Attention:** If the default dump range is used (0 to vmsize-1) and the user's machine is large, or if the user has a large data space or saved segment, the spool space may be depleted, which will cause a system malfunction.
5. Only one message trap can be active. An error message will be issued if a trap is already active and a user enters another SET TRAPMSG command without specifying the REPLACE option.
6. The dump will generate a VMDUMP format spool file when the trap springs. The type of virtual machine being dumped is CMS. The dump can be viewed through the Dump Viewing Facility.
7. The operation of SET TRAPMSG is not dependent on the setting of EMSG and it does not affect the setting of EMSG.
8. This command can also be used to trap messages in the user's application by way of APPLMSG and XMITMSG.
9. This command does not work if the message header was included in the text string by way of APPLMSG with the TEXT or TEXTA option.
10. If either DATASPACE, DCSS or both options are specified, the dump for the data space storage and DCSS storage will be combined into one with the primary dump.
11. Only messages in the CMS system or user repository can be trapped.
12. The STOP option will stop CMS and go to CP when the message trap springs. No dump has been taken yet when going into CP.
13. If you are running in the batch mode, do not set the STOP option; it will be ignored.
14. The default setting for TRAPMSG is OFF.
15. If the dump fails, see DIAGNOSE X'94' in *z/VM: CP Programming Services*.

Limitations of SET TRAPMSG

Only one message can be trapped at a time. In some cases, a group of messages may be received, but only one of the messages can be trapped at once.

Replacing a Message Trap

To replace a message trap, use the same command described above by incorporating the REPlace option.

Turning Off a Message Trap

To turn off a message trap, issue SET TRAPMSG OFF. Additionally, when you re-IPL or log off the virtual machine any previous message trap setting will no longer be in effect.

Querying a Message Trap

To find out whether a message trap has been set, issue the QUERY TRAPMSG command. It will respond with either the message ID that was set with the dump range or a suitable message indicating no message trap is currently set. For more information, see "QUERY TRAPMSG" on page 773.

Obtaining a Dump Triggered by a Message

Once a message trap has been set, the situation that forces the message to occur should be recreated for the Dump to automatically be taken. The dump file will be a standard VMDUMP format spool file placed in the user's or user ID's reader (specified in the command input through the TO option). The dump file can be read into the user's virtual machine with the DUMPLOAD utility. It can then be printed, transferred to another virtual machine, and discarded.

Performance

Because message trap will be checked every time a message is issued from the CMS message facility, system response time may be adversely affected.

Responses

(possible when message trap is triggered)

DMS1297I Dump failed; DIAGNOSE X'94' returns condition code = *cc*; return code = *rc*
DMS1297I Dump has been taken
DMS1297I Dump failed; DIAGNOSE X'94' returns condition code = *cc*; return code = *rc*
DMS1297I Dump has been taken
DMS2047I TRAPMSG dump started; please wait
DMS2047I TRAPMSG dump started for data space: ASIT = *xxxxxxxxxxxxx*; please wait

Messages and Return Codes

- DMS065E *option* option specified twice [RC = 24]
- DMS066E *option* and *option* are conflicting options [RC = 24]
- DMS095E Invalid address *address* [RC = 24]
- DMS149E Userid *userid* not valid [RC = 32]
- DMS771E Invalid message header [RC = 24]
- DMS771E Invalid message number [RC = 24]
- DMS2053E Address range *addr1-addr2* is not completely within your virtual machine [RC = 24]
- DMS2054E Message trap already active; specify REPLACE option [RC = 28]
- DMS2055I STOP option ignored for TRAPMSG command
- DMS2516E Invalid address range: *addr1-addr2*, start greater than end [RC = 24]

SET TVICALL

Authorization

General User

Purpose

Use the SET TVICALL command to control when calls are made to the DMSTVI user exit interface.

Operands

ALL
the CMS OS Simulation routines will call the DMSTVI tape sub-system exit for any valid tape label type.

STD
the CMS OS Simulation routines will call the DMSTVI tape sub-system exit only for the tape standard label types (AL, AUL, SL, SUL).

OFF
the interface to the DMSTVI tape sub-system exit routine has been turned off. No calls for any tape processing will be made to it from the CMS OS Simulation routines.

Initial Setting

TVICALL ALL

Usage Notes

1. The DMSTVI exit is customer or installation provided, it is not part of the CMS product. The DMSTVI exit module is provided by program products that provide tape mounting management facilities such as AMMR. CMS will look for a DMSTVI module when a FILEDEF is issued for a tape device. If it is found, the module will be loaded and a parameter list will be created to call it. The TVICALL status setting can limit how the exit interface may be called during processing.
2. The TVICALL setting will determine whether a SYSPARM verification call will be made to the DMSTVI exit by the FILEDEF command.
3. The TVICALL setting will determine whether the OS Simulation tape handling routines call the DMSTVI exit or use the internal CMS DMSTVS routine to process tapes.
4. If the TVICALL setting is changed during application execution of I/O to tape, the results may be unpredictable.

Messages and Return Codes

System messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

SET UPSI

Authorization

General User

Purpose

Use the SET UPSI command 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.

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 left to right, and any unspecified digits remain unchanged. 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

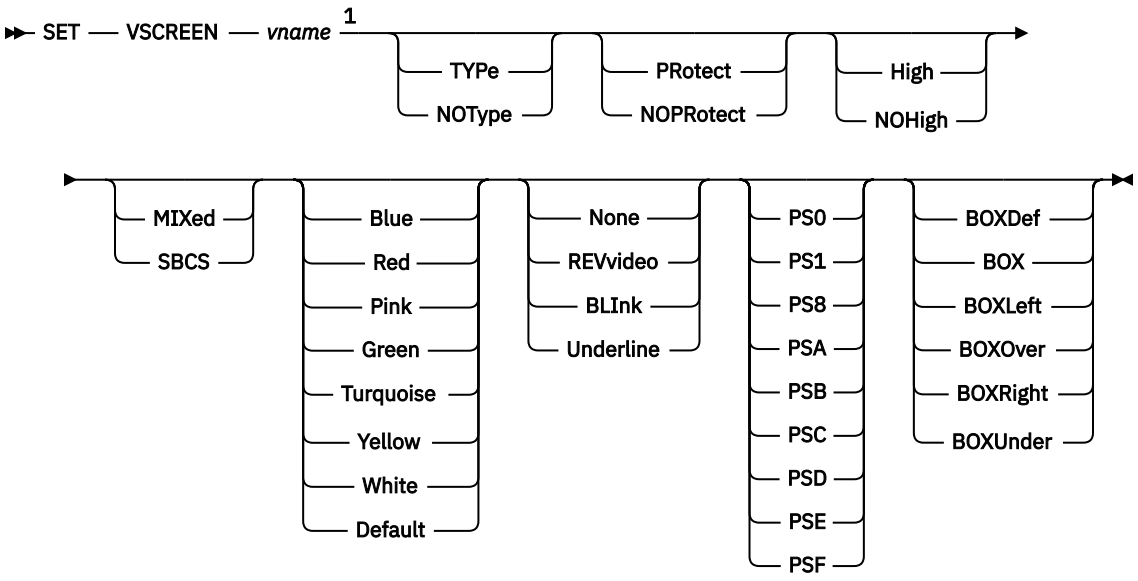
Messages and Return Codes

- DMS099E CMS/DOS environment not active [RC=40]
- DMS109S Virtual storage capacity exceeded [RC=104]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

SET VSCREEN



Notes:

[1] You must specify at least one operand after *vname*.

Authorization

General User

Purpose

Use the SET VSCREEN command to indicate what action should take place when the virtual screen is updated with data.

Operands

vname
is the name of the virtual screen.

TYPe
specifies data is moved to the virtual screen when the virtual screen queue is processed.

NOType
specifies 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.

MIXed
data is displayed in a mixed DBCS (with SO/SI positions) field.

SBCS
data is displayed in a single-byte character set field.

Default
Blue
Red
Pink
Green
Turquoise
Yellow
White
are the choices for field color.

None
REVvideo
BLInk
Underline
are the choices for the extended highlighting of the field. The default is NONE.

PS0
PS1
PS8
PSA
PSB
PSC
PSD
PSE
PSF
are the choices for the programmed symbol set (PSset) of the field. The default is PS0. PS8 specifies a pure DBCS field.

BOXDef
BOX
BOXLeft
BOXOver
BOXRight
BOXUnder
are the choices for the field outlining of the virtual screen for a PS/55-family display. Outlining may be the default outlining for the device, BOX (a full box), or any combination of these to obtain the other possible 14 valid combinations:

BOXLeft
A vertical line on the left of the field

BOXOver
Overline

BOXRight
A vertical line on the right of the field

BOXUnder
Underline

For more information, see "VSCREEN DEFINE" on page 1174.

Initial Setting

For more information, see "VSCREEN DEFINE" on page 1174.

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. For more information, see "SET LOGFILE" on page 965.
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.
3. SET CHARMODE must be ON to display characters using programmed symbol set 1 (PS1).
4. If PS8 is specified, the MIXED and SBCS options have no effect if specified. They are, however, accepted and returned by the QUERY VSCREEN command.
5. Options are accepted whether the device can use them. However, action taken depends on the device capability to support color, extended highlighting, pure DBCS, mixed DBCS, SBCS, and field outlining.
6. Some programs use X'1D' to affect highlighting or color attributes of output. In full-screen CMS X'1D' is a nondisplayable character and does not affect the output.

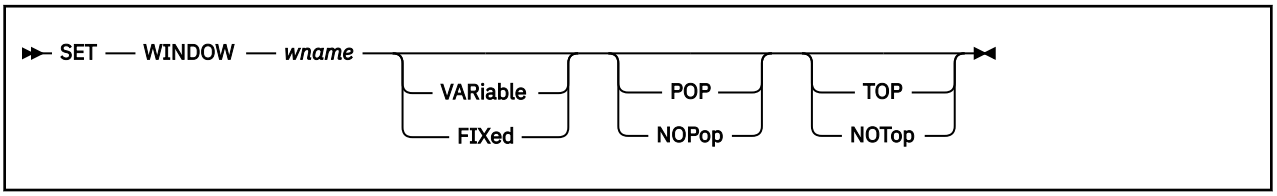
Messages and Return Codes

- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS639E Error in NUCXLOAD routine; return code was *nnn* [RC=24]
- DMS921E Virtual screen *vname* is not defined [RC=28]
- DMS928E Command is not valid for virtual screen *vname* [RC=12]
- DMS3952E Conflicting operand *operand* [RC=24]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	"Command Syntax Error Messages" on page 1411

SET WINDOW



Authorization

General User

Purpose

Use the SET WINDOW command to specify:

- Whether a window is variable or fixed size
- If the window is affected when the virtual screen the window is showing is updated
- Whether the window qualifies as the topmost window.

Operands

wname
is the name of the window.

VARiable
indicates 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 the number of lines in the window is always constant.

POP
specifies the window is displayed on top of all other windows when the virtual screen the window is showing is updated.

NOPop
specifies there is no effect on the window's position in the ordered list of windows when the virtual screen the window is showing is updated.

TOP
specifies 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 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.

Usage Notes

For more information, see “WINDOW DEFINE” on page 1220.

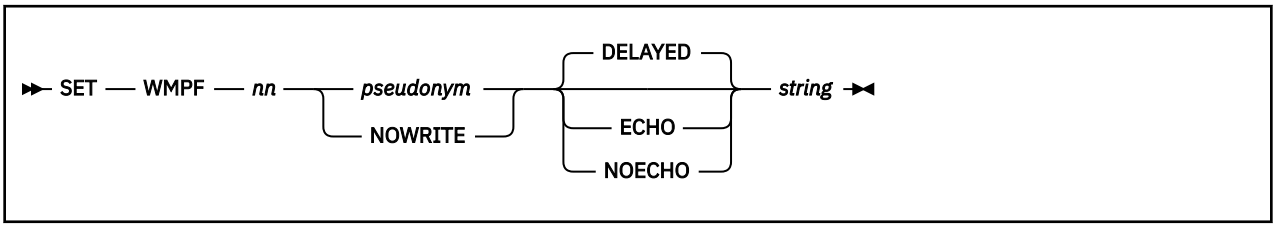
Messages and Return Codes

- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS639E Error in NUCXLOAD routine; return code was *nnn* [RC=24]
- DMS921E Window *wname* is not defined [RC=28]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

SET WMPF



Authorization

General User

Purpose

Use the SET WMPF command to set a WMPF key to execute a windowing command.

Operands

nn
is a number from 1-24 indicating which PF key is being set.

pseudonym
is a 1-9 character representation of the PF key definition. The pseudonym is displayed in the PF key definition area at the bottom of the CMS window. Mixed DBCS data can be used for *pseudonym*, provided your terminal is capable of supporting mixed DBCS.

NOWRITE
suppresses overwriting of the PF key pseudonym when you set a WMPF key.

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 Enter. 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 a string, or command(s) to be executed when the key is pressed.

Initial Setting

```
WMPF 01 Help        NOECHO   HELP
WMPF 02 Top         NOECHO   WINDOW TOP =
WMPF 03 Quit        NOECHO   WINDOW DROP WM
WMPF 04 Clear       NOECHO   WINDOW CLEAR =
WMPF 05 Copy        NOECHO   PSCREEN PUT COPY SCREEN
WMPF 06 Retrieve             RETRIEVE
WMPF 07 Backward    NOECHO   WINDOW BACKWARD = 1
WMPF 08 Forward     NOECHO   WINDOW FORWARD = 1
WMPF 09 Maximize    NOECHO   WINDOW MAXIMIZE =
```

```
WMPF 10 Left        NOECHO   WINDOW LEFT = 10
WMPF 11 Right       NOECHO   WINDOW RIGHT = 10
WMPF 12 Restore     NOECHO   WINDOW RESTORE =
```

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:

CP	SET RESERVED	WINdow MINimize
HELP	SET WINDOW	WINdow Next
PSCreen PUT	SET WMPF	WINdow POP
QUERY BORDER	WINdow BAckward	WINdow POSition
QUERY HIDE	WINdow Bottom	WINdow REStore
QUERY LOCATION	WINdow CLEar	WINdow Right
QUERY RESERVED	WINdow Down	WINdow SHOw
QUERY SHOW	WINdow DROp	WINdow SIZe
QUERY WINDOW	WINdow Forward	WINdow Top
QUERY WMPF	WINdow HIDe	WINdow Up
SET BORDER	WINdow Left	
SET LOCATION	WINdow MAXimize	

   In the WM environment, you can enter HELP (WMPF 1) to see the list of available commands. The WM environment creates a WMHELP window and WMHELP virtual screen to display the list. To exit the WM window, use the WINDOW DROP command (WMPF 3).
2. To cancel a PF key definition (resetting it to no operation), 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, 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 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

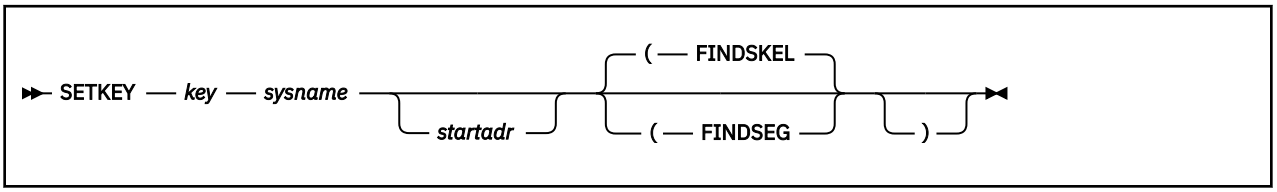
- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS525E Invalid PFkey number [RC=24]

- DMS639E Error in NUCXLOAD routine; return code was *nnn* [RC=24]
- DMS926E 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 their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

SETKEY



Authorization

General User

Purpose

Use the SETKEY command to change CMS storage keys.

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 discontiguous 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.

if *startadr* isn't specified, the storage key is assigned starting with the first page in the segment. This is determined by issuing a DIAG code X'64' FINDSPACE or FINDSKEL (depending on what option you have specified).

Options

For an explanation of active and skeleton segment searches, see *z/VM: CP Programming Services*.

FINDSKEL
bypasses any search for active segments and searches for skeleton segments (an inactive segment definition for which a class S SDF spool file has been created). If the FINDSKEL fails, a FINDSEG is issued. This is the default.

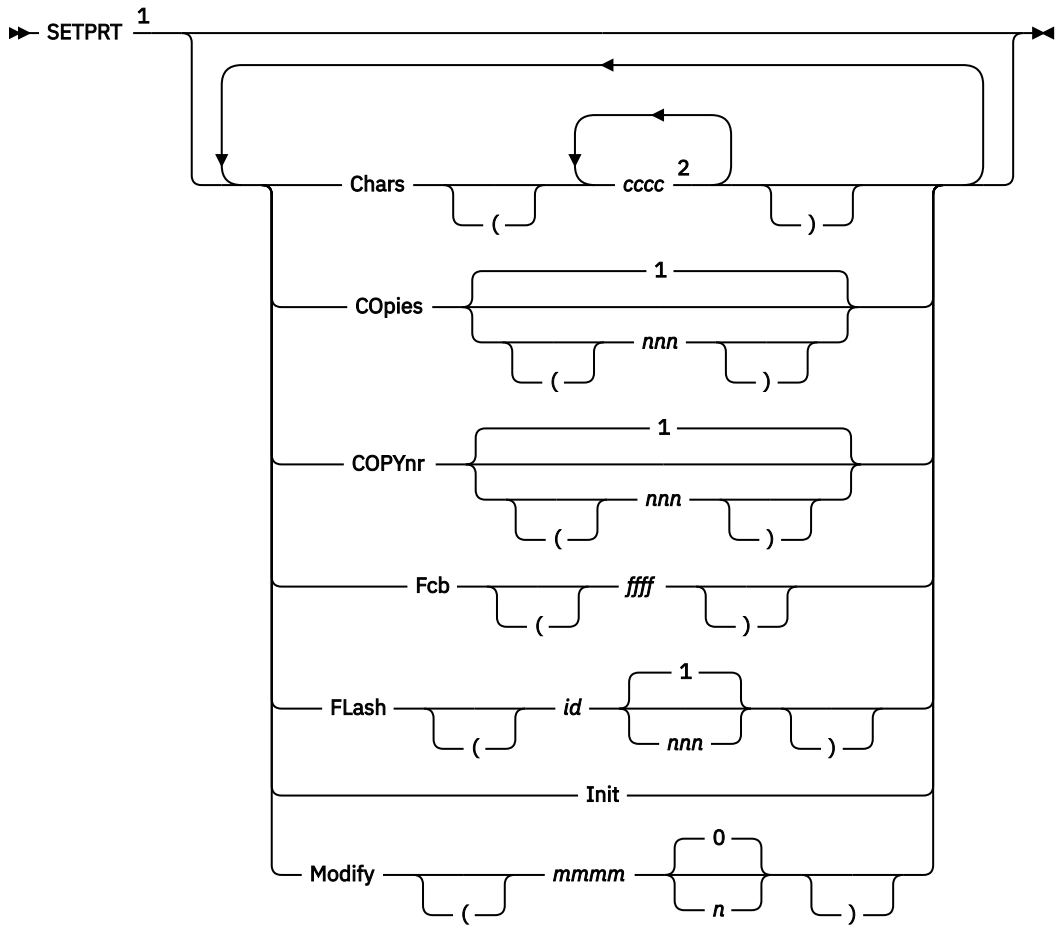
FINDSEG
finds the saved segment by first searching for an active segment, and then for a skeleton segment (FINDSKEL) if no active segment exists.

Messages and Return Codes

System messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

SETPRT



Notes:

[1] Opening parenthesis optional; closing parenthesis necessary only if multiple keywords are entered, or a keyword value is entered that could be interpreted as a SETPRT keyword or keyword abbreviation.

[2] Specify a maximum of four times.

Authorization

General User

Purpose

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.

Operands

Chars *cccc*

specifies the names of from 1-4 character arrangement tables (CATs) to be loaded into the virtual 3800 printer. CAT names may be from 1-4 alphanumeric 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 1-255. 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 1-255. 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 1-4 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 0-255. If *nnn* is not specified, the default is 1. If the FLASH keyword is omitted, no copies are printed with an overlay.

Init
specifies 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, the default is 0.

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 with the CP GENIMAGE command. You cannot issue GENIMAGE with the base file ID and then use the alias with SETPRT.

   The name of the alias must conform to the naming conventions of the TEXT file. For example, an FCB must exist as FCB3*ffff* TEXT.
2. System interfaces for printing (like PRINT and PRINTL) allow you to indicate data being printed contains a Table Reference Character (TRC) byte. This byte identifies which Character arrangement table (CAT) to use when printing the file. The CATs contain different fonts. For more information on the names of the CATs, see *z/VM: CP Commands and Utilities Reference*. Any CATs must be specified so they correspond to the appropriate Table Reference Character (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, because '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.

Examples

Example 1: For example, to indicate you want to use character set GF10 printed at 6 lines per inch, enter:

```
setprt chars gf10 fcb 6
```

Example 2: The following are all valid SETPRT commands:

```
SETPRT Chars (AN
SETPRT Chars (AN AOA
SETPRT Chars (AN AOA) COpies (1)
SETPRT Chars  AN AOA  COpies  1
SETPRT Chars  AN AOA)
SETPRT Chars  AN AOA) COpies 1
```

The following is not a valid SETPRT command, because COPIES will be used as a value for CHARS, which the system does not allow.

```
SETPRT Chars (AN AOA  COpies (1)
```

The following is also not a valid, if INIT is intended to be used as a keyword; it is valid only if INIT is intended as a value for CHARS.

```
SETPRT Chars (AN AOA Init
```

Responses

```
DMSSPR196I  Printer vdev setup complete
```

The virtual 3800 printer was successfully loaded.

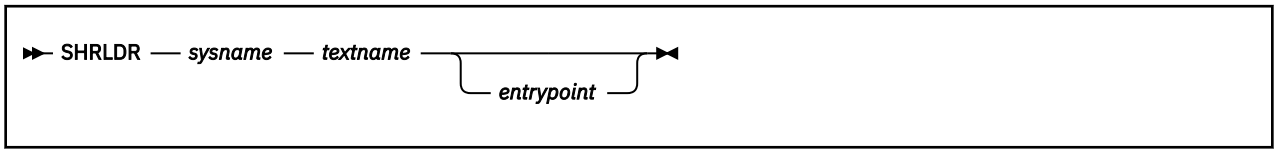
Messages and Return Codes

- DMS002E File [*fn [ft [fm]*]] not found [RC=28]
- DMS014E Invalid keyword *function* [RC=24]
- DMS026E Invalid value *value* for *keyword* keyword [RC=24]
- DMS113S Printer 00E not attached [RC=100]
- DMS145S Intervention required on printer [RC=100]
- DMS197S Undiagnosed error from printer 00E [RC=100]
- DMS198E SETPRT load check; sense=*sense*
- DMS199S Printer 00E not a virtual 3800 Model 1 or Model 3
- DMS204E Too many WCGMs needed for CHARS
- DMS257T Internal system error at address *address* (offset *offset*)
- DMS352E Invalid SETPRT data in file *fn ft*
- DMS1144E Implicit rollback occurred for work unit *workunitid* [RC=31]
- DMS1252T Rollback unsuccessful for file pool *filepoolid*

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in the Shared File System	“File Pool Server Messages” on page 1414
Errors in using a file	“File Error Messages” on page 1418

SHRLDR



Authorization

General User

Purpose

Use the SHRLDR command to load a saved segment in nonshared mode.

Operands

sysname
is the name of the saved segment to be loaded. This is the segment name defined by the CP DEFSEG command. See the DEFSEG command in *z/VM: CP Commands and Utilities Reference* for more information.

textname
is the name of the text file to be loaded in the *sysname* saved segment. The file type must be TEXT, and this file can be on any accessed minidisk or shared file system (SFS) directory.

entrypoint
is the name of a specific entry point within the text deck. This is the code that is given control at run time. If the first CSECT address is not the entry point you want given control at run time, this must be specified.

Usage Notes

1. To load a nonshared saved segment, a NAMESAVE entry must be included in the user directory for each saved segment you load.
2. To use SHRLDR to save a saved segment from a virtual machine, do the following:
   a. Create a skeleton system data file for the saved segment using the CP DEFSEG command.
   b. Make sure your virtual storage encompasses the entire segment, then use SHRLDR to load the segment.
   c. Set the storage keys using the SETKEY command. See "SETKEY" on page 1014 for more information.
   d. Save the saved segment using the CP SAVESEG command.

   For more information on the DEFSEG and SAVESEG commands, see *z/VM: CP Commands and Utilities Reference*. For more information about saved segments, see *z/VM: Saved Segments Planning and Administration*.
3. The specified saved segment is accessed and the text file is then loaded at the saved segment's starting address.
4. Only a single text file is acceptable as input. If the system to be saved consists of multiple text files, use the PRELOAD command to create a single text file. See *z/VM: VMSES/E Introduction and Reference* for information about the PRELOAD command.
5. If the text file resides in a shared file system (SFS) directory, that directory must be accessed. If it is not already accessed, use the CMS ACCESS command to access it.
6. A single line, which describes the results of the operation, is stacked by the program. The format of the stacked line is:

Token
Content

1
* (viewed as a comment by CMS)

2
saved segment name

3
saved segment load address

4
saved segment end address

5
entry point address

Messages and Return Codes

- DMS021E Entry point *name* not found
- DMS104S Error *nn* reading file *fn ft fm* from disk or directory [RC=*xx*]
- DMS283E The *name* saved segment could not be found; return code *rc* from SEGMENT [RC=*xx*]
- DMS283E The *name* saved segment could not be loaded; return code *rc* from SEGMENT [RC=*xx*]
- DMS374W Zero-length CSECT *csect* encountered [RC=1]
- DMS1262S Error *nn* opening file *fn ft fm* [RC=*xx*]
- DMS2158I START: *addr*, END: *addr*, #PAGES: *n*
- DMS2161E Text load address does not match segment start address. [RC=1]
- DMS2161E Invalid text data. [RC=1]
- DMS2161E Control section *name* too large: *addr* [RC=1]
- DMS2161E Relocated address constant does not fall within the segment definition.[RC=1]

Return codes:

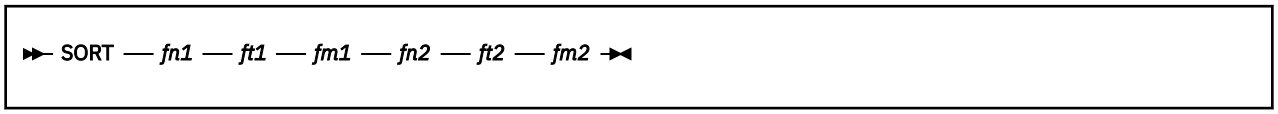
RC
Meaning

1
Invalid load data

xx
Error reading file (RC from FSREAD)

xxx
Segment find/load error (RC from SEGMENT)

SORT



Authorization

General User

Purpose

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.

Operands

fn1 ft1 fm1
is the name of the file containing the records to be sorted.

fn2 ft2 fm2
is the name of the new output file to contain the sorted records.

Usage Notes

1. The input and output files must not have the same file identifiers, because SORT cannot write the sorted output back into the space occupied by the input file. If *fileid2* is the same as *fileid1*, message

   ```
   DMSSRT019E 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.
2. Entering Sort Control Fields: After the SORT command is entered, CMS responds with the following message on the terminal:

   ```
   DMSSRT604R  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.
3. 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.

Responses

```
DMSSRT604R   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."

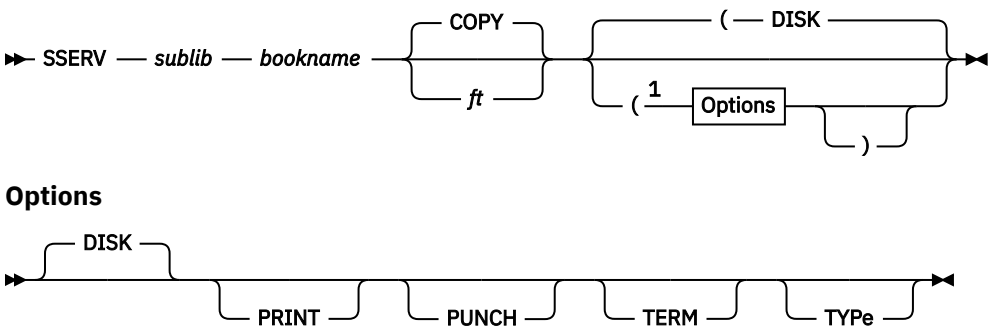
Messages and Return Codes

- DMS002E File [*fn [ft [fm]*]] not found [RC=28]
- DMS009E Column *col* exceeds record length [RC=24]
- DMS019E Identical fileids [RC=24]
- DMS034E File *fn ft fm* is not fixed length [RC=32]
- DMS037E Filemode *mode*[(*vdev*)] is accessed as read/only [RC=36]
- DMS053E Invalid sort field pair defined [RC=24]
- DMS054E Incomplete fileid specified [RC=24]
- DMS062E Invalid * in fileid *[fn ft [fm]]* [RC=20]
- DMS063E No list entered [RC=40]
- DMS069E Filemode *mode* not accessed [RC=36]
- DMS070E Invalid parameter *parameter* [RC=24]
- DMS104S Error *nn* reading file *fn ft fm* from disk or directory [RC=100]
- DMS105S Error *nn* writing file *fn ft fm* on disk or directory [RC=100]
- DMS212E Maximum number of records exceeded [RC=40]
- DMS2521E SORT cannot be performed on empty file *fileid1* [RC=88]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411
Errors in the Shared File System	“File Pool Server Messages” on page 1414
Errors in using a file	“File Error Messages” on page 1418

SSERV



Notes:

[1] You can enter Options in any order between the parentheses.

Authorization

General User

Purpose

Use the SSERV command in CMS/DOS to copy, display, print, or punch a book from a VSE source statement library.

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. If a file type is not specified, *ft* defaults to COPY. The file name is always the same as the book name.

Options

You may enter as many options as you choose, depending upon the functions you want to perform. If you specify more than one option, CMS uses the last option specified.

DISK
copies the book to a CMS file. This is the default.

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.

TYPe
displays the book on your terminal. (This option has the same function as the TERM option.)

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.
2. You should not use the SSERV command to copy books from macro (E) sublibraries, because 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

- DMS003E Invalid option: *option* [RC=24]
- DMS004E Book *subl.book* not found [RC=28]
- DMS006E No read/write A filemode accessed [RC=36]
- DMS070E Invalid parameter *parameter* [RC=24]
- DMS089E Open error code *nn* on SYSSLB [RC=36]
- DMS097E No SYSRES volume active [RC=36]
- DMS098E No book name specified [RC=24]
- DMS099E CMS/DOS environment not active [RC=40]
- DMS105S Error *nn* writing file *fn ft fm* on disk or directory [RC=100]
- DMS113S Disk(*vdev*) not attached [RC=100]
- DMS194S Book *subl.book* contains bad records [RC=100]
- DMS411S Input error code *nn* on SYS*aaa* [RC=*rc*]

STAG

Purpose

Use the STAG command to stack the tag information associated with a specified reader file.

Operands

spoolid
is the spool ID of a file in the virtual reader, which must be given as exactly four decimal digits. If not specified, the tag data for the first reader file is stacked.

Usage Notes

1. To associate file descriptive information with a z/VM spool file, use the CP TAG command. For details, see *z/VM: CP Commands and Utilities Reference*.
2. In addition to the STAG command, you can also use the following CMS command to obtain similar results. See "EXECIO" on page 229 for details.

```
EXECIO * CP (STRING TAG QUERY FILE spoolid
```

Messages and Return Codes

- DMS070E Invalid parameter *parameter* [RC=3]
- DMS655E Spoolid *nnnn* does not exist [RC=2]
- DMS2162I Specified file has no tag data [RC=1]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	"Command Syntax Error Messages" on page 1411

Return codes:

RC
Meaning

0
Operation complete — tag data stacked

1
Minidisk or directory is read-only (including read-only extensions)

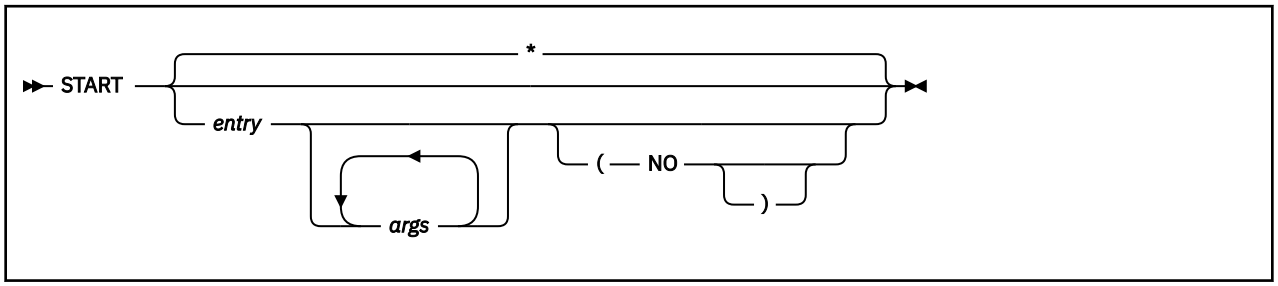
2
Mode is not in use (nothing stacked)

3
Invalid parameter list (nothing stacked)

xx
where xx is RC from PARSECMD or CMSCALL

100
Explanation complete (when '?' specified)

START



Authorization

General User

Purpose

Use the START command to begin execution of CMS, OS, or VSE programs previously loaded or fetched.

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 with a control section name or an entry point name.

***
passes control to the default entry point. For more information on how this point is selected, see "LOAD" on page 471.

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 by means of 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.

Options

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 generated through the GENMOD command with the NOMAP option or for modules generated by the BIND command. The START command does not execute properly for such programs.
3. 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
```

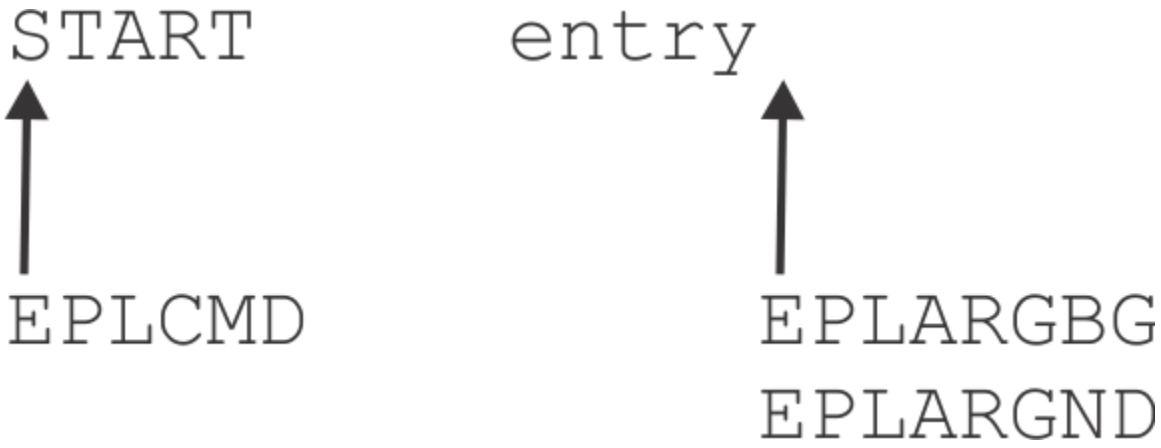
For more information, see the appropriate language guide on the parameter requirements.

4. 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.
5. 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:

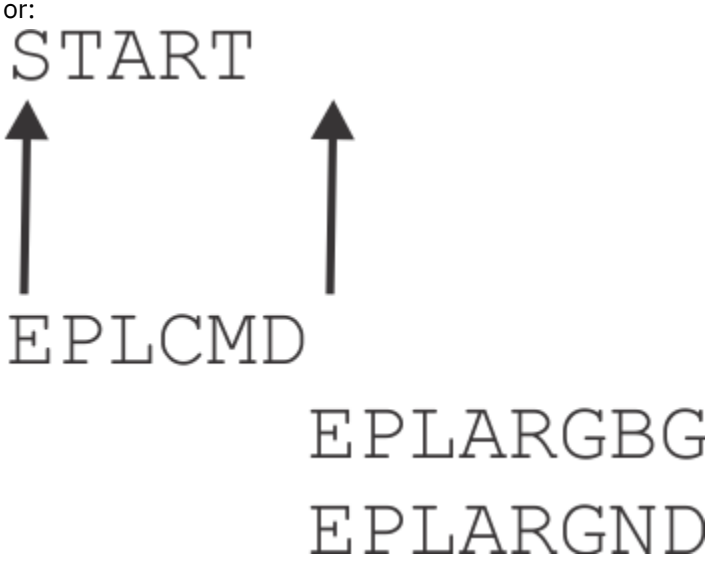
```
DC     A(EPLCMD)
DC     A(EPLARGBG)
DC     A(EPLARGND)
DC     A(0)
```

Where:

or:



or:



Responses

```
DMSLIO740I  Execution begins ...
```

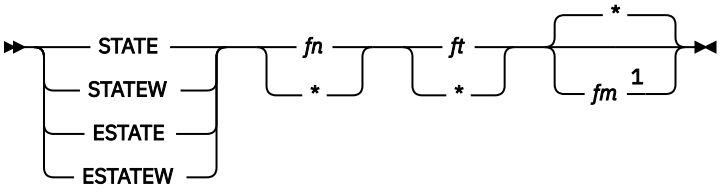
is displayed when the designated entry point is validated.

This message is suppressed if CMS/DOS is active and the COMP option is specified in the FETCH command.

Messages and Return Codes

- DMS021E Entry point *name* not found [RC=40]
- DMS055E No entry point defined [RC=40]

STATE/STATEW (ESTATE/ESTATEW)



Notes:

[1] File mode number specified is ignored when both the file name and file type are specified; it is used when either the file name or file type are specified as asterisks.

Authorization

General User

Purpose

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 (for which you have write authority).

It may be necessary to use ESTATE/ESTATEW when writing assembler programs. This enables STATE to handle files larger than 65535 records. This is the equivalent of coding the FSSTATE macro with the FORM=E option. For more information on the FSSTATE macro, see *z/VM: CMS Macros and Functions Reference*.

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/ESTATEW command specifying a file that exists on a read-only disk or directory, you receive error message DMS002E or file not found.
2. For data stored in SFS directories, STATE will not find subdirectories, erased or revoked aliases, external objects, 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.

5. 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.

6. 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.
7. You can invoke the STATE/STATEW (ESTATE/ESTATEW) command from the terminal, from an exec file, or as a function from a program. If STATE/STATEW (ESTATE/ESTATEW) is invoked as a function or from an exec file that has the message output suppressed, messages DMS002E, DMS069E and DMS070E are not issued.
8. If the STATE/STATEW (or ESTATE/ESTATEW) command is invoked by SVC 204 in an assembler program, the address of the STATEFST copy is returned at X'1C' into the STATE parameter list.
9. Using the FORCERW option on the ACCESS command, you can access another user's SFS directory in read/write status. If you issue the STATEW (ESTATEW) command for a file in that directory, STATEW (ESTATEW) will find the file only if you have write authority to it. Otherwise, you will receive error message DMS002E (file not found).
10. If you access one of your own SFS directories in read/write status, the STATEW (ESTATEW) command will find only files for which you have write authority. Suppose, for example, you have an alias to another user's file. If you have read authority to the base file, STATEW (ESTATEW) will not find the file. If you have write authority, however, STATEW (ESTATEW) will find the file.
11. The STATE/STATEW (ESTATE/ESTATEW) command will find an open, empty file (minidisk or SFS) if it was opened using OS Open or FS Open interfaces. However, the LISTFILE command will not find an open, empty file.
12. The STATE command will not find new files opened and written to by REXX I/O interfaces (for example: LINEOUT, CHAROUT) or by the DMSOPEN, and DMSWRITE CSL routines until they are closed, you will receive error message DMS002E or file not found.

Responses

The CMS ready message indicates the specified file exists.

```
DMS227I  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.

```
DMS228I  User labels bypassed on data set
data set name
```

The specified data set has disk user labels; these labels are skipped.

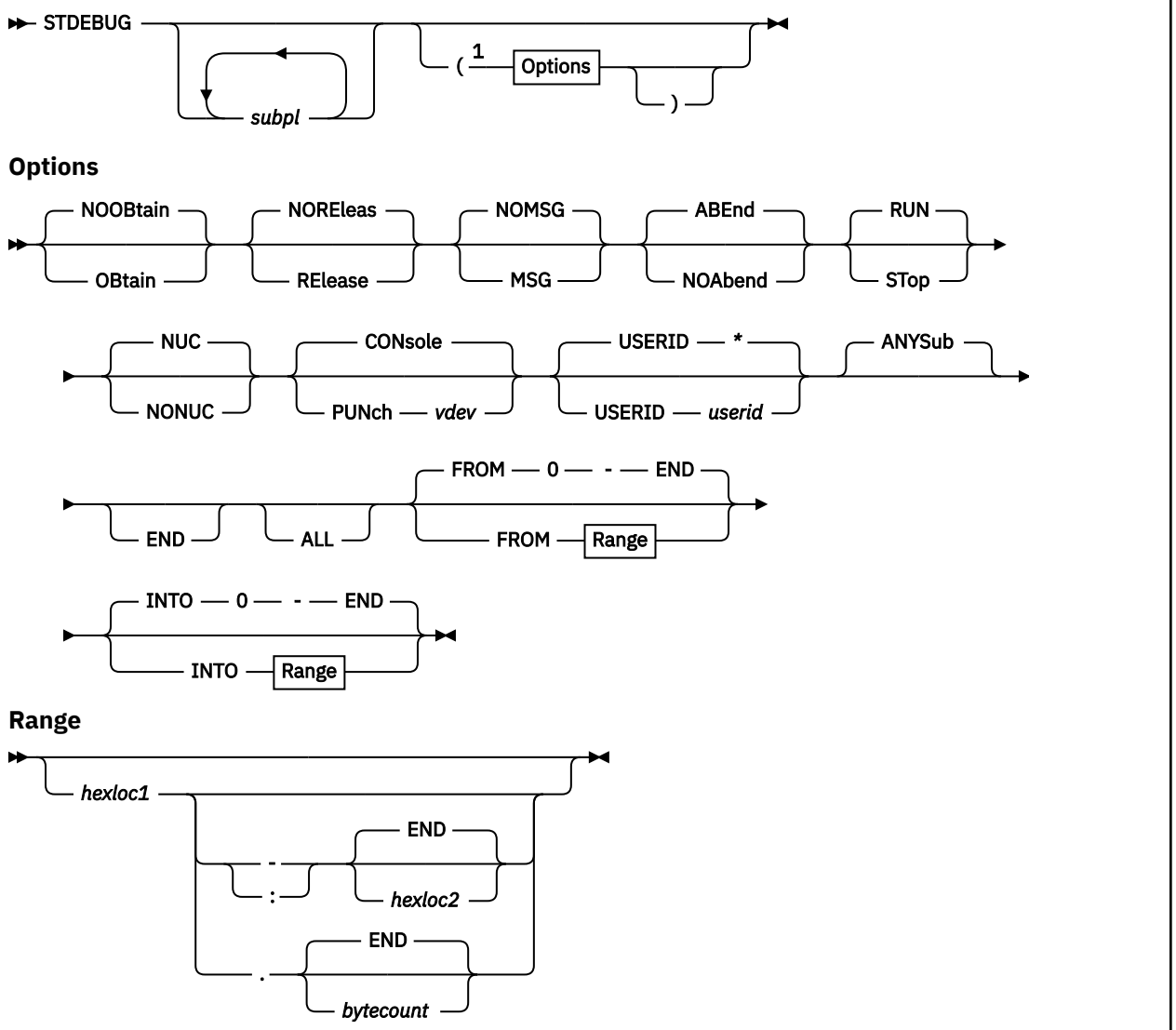
Messages and Return Codes

- DMS002E File [*fn [ft [fm]*]] not found [RC=28]
- DMS048E Invalid filemode *mode* [RC=24]
- DMS054E Incomplete fileid specified [RC=24]
- DMS062E Invalid character *char* in fileid *fn ft* [RC=20]
- DMS062E SO and SI are invalid fileid characters [RC=20]
- DMS069E Filemode *mode* not accessed [RC=36]
- DMS070E Invalid parameter *parameter* [RC=24]
- DMS229E Unsupported OS data set, error *nn* [RC=8*n*]
- DMS253E 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 their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

STDEBUG



Notes:

[1] You can enter Options in any order between the parentheses.
All defaults are for the initial invocation and remain in effect unless changed in a later invocation.

Authorization

General User

Purpose

Use the STDEBUG command to monitor CMS storage management activity (OBTAIN and RELEASE requests) within your virtual machine. You can invoke STDEBUG multiple times, with each invocation building on the previous one. For more information, see Usage Note “3” on page 1035.

Operands

subpl
specifies you want storage calls for storage in this subpool or these subpools to be traced.

Subpool names can be from 1-8 characters, cannot contain blanks or nondisplayable characters, and are case sensitive. Use the ALL option to display information on subpools with names you can't directly specify. For subpool names containing nonblank, nonprintable characters, pass the names from a program using the extended plist. For more information about how to do this, see *z/VM: CMS Application Development Guide for Assembler*.

Options

OBtain
applies the traps specified by INTO, FROM, and the list of subpool names to requests to obtain free storage such as:

- CMSSTOR OBTAIN
- DMSFREE
- GETMAIN

NOOBtain
does not apply the traps specified by INTO, FROM, and the list of subpool names to OBTAIN requests.

RElease
applies the traps specified by INTO, FROM, and the list of subpool names to requests to release free storage such as:

- CMSSTOR RELEASE
- DMSFRET
- FREEMAIN

Whenever a SUBPOOL RELEASE or SUBPOOL DELETE is issued, the UNALLOC function is displayed instead, indicating how the storage was released.

NOREleas
does not apply the traps specified by INTO, FROM, and the list of subpool names to RELEASE requests.

NOMSG
suppresses storage management error messages if the caller to storage management specified MSG=NO on the macro invocation. NOMSG is the default on the *initial* invocation of STDEBUG.

MSG
displays all suppressed error messages, even if the caller to storage management specified MSG=NO.

ABEnd
performs ABEND processing on error conditions when ERROR= ABEND was specified on the macro invocation. ABEND is the default on the *initial* invocation of STDEBUG.

NOAbend
means do not perform ABEND processing on error conditions even if ERROR=ABEND was specified on the macro invocation. Instead, return control to the caller as if ERROR=* had been specified on the invocation.

RUN
continues processing generally, and does not place the virtual machine into CP READ after storage management messages have been issued. RUN is the default on the *initial* invocation of STDEBUG.

STop
places the virtual machine into CP READ after a Storage Management trace or error message has been displayed.

If you specify STOP without specifying MSG, you may go into CP READ with no indication why. This is because the caller to storage management specified MSG=NO on the call, and you did not specify

MSG on STDEBUG to override that setting. Therefore the message is suppressed when you enter CP read, as requested by STOP.

NUC
includes calls from the CMS nucleus in the trace. These calls are bounded by NUCALPHA on the low end and NUCOMEGA on the high end, and they must also fall within the specified FROM range. NUC is the default on the *initial* invocation of STDEBUG.

NONUC
does not include calls from the CMS nucleus in the trace, even if they fall within the specified FROM range.

CONsole
displays trace messages as CP messages. CONsole is the default on the *initial* invocation of STDEBUG.

Note: STDEBUG uses CP messages for output. Do **not** use the console as the output device when an application connected to the *MSG or *MSGALL CP system services is executing. The external interrupts caused by the CP messages generated by STDEBUG in themselves cause calls to Storage Management, which are then trapped and cause more messages to be issued.

Direct messages to an alternate user ID with the USERID option or to another device with the PUNCH option.

STDEBUG does not issue messages to the console when SET FULLSCREEN is ON, unless they are being directed to a different user ID (with the USERID option) than the one being traced.

PUNch *vdev*
writes the trace data to the unit record output device specified by *vdev*.

The device specified by *vdev* can be any valid unit record output device; it does not have to the standard CMS devices, such as X'D' for the punch or X'E' for the printer. This allows you to use devices that do not conflict with those being used by the application you are tracing.

USERID *userid*
directs the CP messages, generated when the CONSOLE option is specified, to the alternate user ID specified by *userid*. The initial value of USERID is the virtual machine invoking STDEBUG. You can reset it to this original value any time by issuing:

```
STDEBUG (USERID *
```

or:

```
STDEBUG (END
```

FROM *range*
traps only the calls to storage management where the address of the call is within the specified range. (If any byte of an address is within the range, the address is considered to be within the range.)

A range may be a single address, a pair of addresses separated by a "-" or ":", or an address followed by a period and a byte count. You may also specify END after the delimiter to include storage from the starting address to the end of the virtual machine.

hexloc1
is the starting address of the range.

hexloc2
is the ending address of the range.

bytecount
is the number of bytes to be mapped.

The values *hexloc1*, *hexloc2*, and *bytecount* must be from 1-8 hexadecimal digits. Leading zeros are optional.

Blanks are not allowed within a range specification, but each range must be separated from the other ranges by one or more blanks. X'0-END' is the default on the *initial* invocation of STDEBUG.

INTO *range*
traps only the calls to Storage Management where the storage being obtained or released falls within the specified range. For more information on the valid ranges, see the preceding description of the FROM operand. X'0-END' is the default on the *initial* invocation of STDEBUG.

ANYSub
resets a previous specification of a list of subpool names, and base the tracing of obtain and release calls on the current setting of FROM and INTO for any subpool. ANYSUB is the default on the *initial* invocation of STDEBUG.

END
ends and resets all tracing of calls to Storage Management.

ALL
traces all calls to storage management for any subpool or address range. Specifying STDEBUG (ALL is the same as specifying:

```
STDEBUG (OB RE ANYSUB RUN MSG NUC CON INTO 0-END FROM 0-END
```

Usage Notes

1. STDEBUG installs itself as a nucleus extension the first time it is invoked. It may thereafter be invoked as an immediate command.
2. Calls invoked from assembler language programs by SVC 202 or CMSCALL must provide both an extended and tokenized plist. If the extended plist is not available, an error will occur.
3. STDEBUG may be invoked multiple times. Each time, previously specified options may be changed or new options may be specified that are added to the previous options.

   Specifying an option over its opposite on a previous invocation overrides the previous setting.

   Specifying an option over the same setting on a previous invocation maintains the setting.

   The END and ALL options also set other options to certain values invoked on top of previous invocations, resetting them if necessary:

   END
   sets values NOOBTAin, NORELEase, NOMSG, ABEND, RUN, NUC, CONSOLe, USERID, FROM, INTO, and ANYSUB.

   ALL
   sets values OBTAin, RELEase, MSG, RUN, NUC, CONSOLe, FROM, INTO, and ANYSUB.

Examples

The following example shows 5 invocations of STDEBUG, one after another, with explanation in each step of how each subsequent invocation builds on or changes the previous one:

1. ```
   STDEBUG (OB
   myprog
   ```

   The first call traces only obtain requests. As this is the first invocation of STDEBUG, the tracing would be for storage in the address range of X'0-END' and a caller range of X'0-END'. In addition, the tracing would be to the virtual console and the user ID is the invoker of STDEBUG. It traces any subpool.

2. ```
   STDEBUG (RE INTO 20000-30000
   myprog
   ```

   The second call does not change the tracing of obtain requests and adds the tracing of release requests. It also limits tracing to only storage within the specified address range.

3. ```
   STDEBUG NUCLEUS (PUN D
   myprog
   ```
   ```

The third call does not change anything previously set, but it restricts the tracing to only calls for the NUCLEUS subpool. In addition, the tracing is now directed to the punch rather than the console.

4.
```
STDEBUG USER XYZ
myprog
```

The fourth call traces the USER and XYZ subpools instead of the NUCLEUS subpool. All tracing of the NUCLEUS subpool ends. The previous punch and address specifications remain in effect.

5.
```
STDEBUG (ANYSUB CON INTO 0-END
myprog
```

The fifth call resets the tracing to any subpool rather than just the USER and XYZ subpools, and would direct the tracing back to the console rather than the punch. In addition, the address range specified traces storage within the entire virtual machine.

Responses

1. Format of the trace information directed to the virtual console with the CONSOLE option:

```
                          (1)      (2)      (3)      (4)      (5)
08:40:07 * MSG FROM FRODO: OBTAINED 00000510 013FEA38 DMSXEDIT 00ED3D9E
08:40:07 * MSG FROM FRODO: OBTAINED 000002A8 013FB000 DMSXEDIT 00EBC19E
08:40:07 * MSG FROM FRODO: OBTAINED 00000168 013FBE98 DMSXEDIT 00EBC27E
08:40:07 * MSG FROM FRODO: OBTAINED 00003DD8 013EA000 DMSXEDIT 00EC8CE6
08:40:07 * MSG FROM FRODO: OBTAINED 00000800 013FF6C8 DMSBLOKN 00E2AE78
08:40:07 * MSG FROM FRODO: RELEASED 00000800 013FF6C8 DMSBLOKN 00E28F0C
08:40:16 * MSG FROM FRODO: UNALLOC  00001000 00DED000 DMSXEDIT 00EBE9D6
08:40:16 * MSG FROM FRODO: UNALLOC  00001000 013FE000 DMSXEDIT 00EBE9D6
```

Where:

(1)
is the name of the event: OBTAINED, RELEASED, or UNALLOC

(2)
is the number of bytes obtained or released

(3)
is the address of the storage being obtained or released

(4)
is the name of the subpool that owns the storage

(5)
is the address of the caller to storage management

2. Format of the trace information directed to a unit record device with the PUNCH option:

```
08:52:09 OBTAINED BYTES=000000B8 ADDR=013FEF48 SUBPL=DMSXEDIT CALLER=00EBA162
08:52:09 OBTAINED BYTES=00000780 ADDR=013FC880 SUBPL=DMSXEDIT CALLER=00EBD846
08:52:09 OBTAINED BYTES=00000510 ADDR=013FEA38 SUBPL=DMSXEDIT CALLER=00ED3D9E
08:52:09 OBTAINED BYTES=000002A8 ADDR=013FB000 SUBPL=DMSXEDIT CALLER=00EBC19E
08:52:09 OBTAINED BYTES=00000168 ADDR=013FBE98 SUBPL=DMSXEDIT CALLER=00EBC27E
08:52:09 OBTAINED BYTES=00003DD8 ADDR=013EA000 SUBPL=DMSXEDIT CALLER=00EC8CE6
08:52:09 OBTAINED BYTES=00000800 ADDR=013FF6C8 SUBPL=DMSBLOKN CALLER=00E2AE78
08:52:09 RELEASED BYTES=00000800 ADDR=013FF6C8 SUBPL=DMSBLOKN CALLER=00E28F0C
08:52:10 OBTAINED BYTES=00000008 ADDR=013FFEC0 SUBPL=DMSBLOKN CALLER=00EDA550
08:52:10 RELEASED BYTES=00000008 ADDR=013FFEC0 SUBPL=DMSBLOKN CALLER=00E0E47E
08:52:10 UNALLOC  BYTES=00001000 ADDR=00DED000 SUBPL=DMSXEDIT CALLER=00EBE9D6
08:52:10 UNALLOC  BYTES=00001000 ADDR=013FE000 SUBPL=DMSXEDIT CALLER=00EBE9D6
```

Messages and Return Codes

- DMS389E Invalid device address: *device* [RC=24]
- DMS2513E Extended plist is required. [RC=24]
- DMS2516E Invalid address range: *addr1-addr2*, start greater than end. [RC=24]
- DMS2518E Error, RC=*nn* from STDEBUG initialization[RC=*rc*]

- DMS8503E Invalid STDEBUG parameter list. [RC=24]

Return codes:

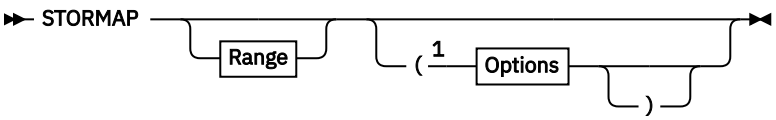
RC
Meaning

99
STDEBUG was invoked on an unsupported level of CMS.

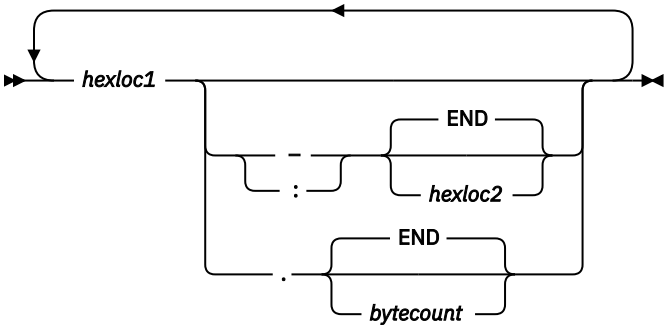
Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

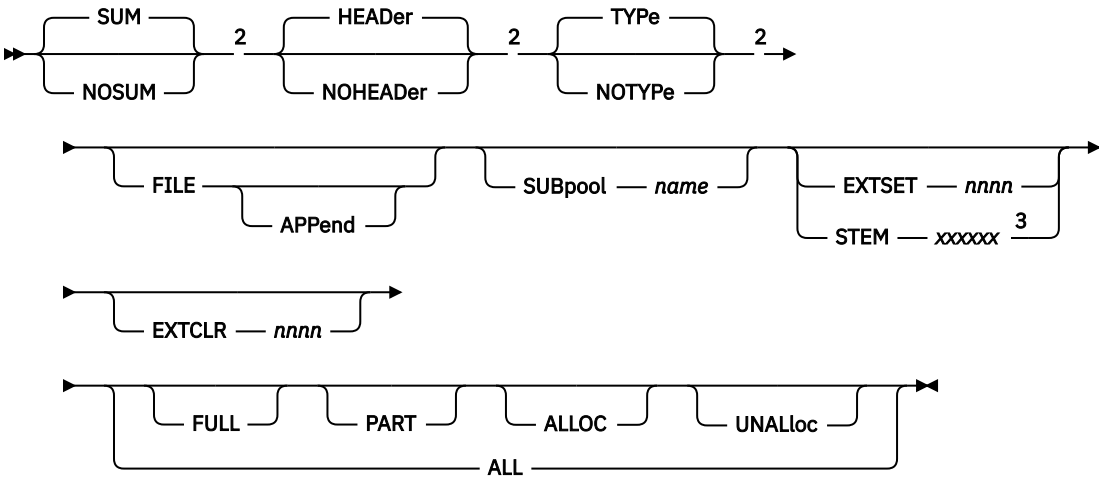
STORMAP



Range



Options



Notes:

1 You can enter Options in any order between the parentheses.
2 This default can change when other options are specified.
3 For use with REXX and EXEC 2 only.

Authorization

General User

Purpose

Use the STORMAP command to map the storage within your virtual machine.

Operands

range
specifies one or more address ranges to be included in the map. (If any part of a storage block falls within the range, the storage block is considered to be within the range.) A range may be a single address, a pair of addresses separated by a "-" or ":", or an address followed by a period and a byte count. You may also specify END after the delimiter to include storage from the starting address to the end of the virtual machine.

hexloc1
is the starting address of the range.

hexloc2
is the ending address of the range.

bytecount
is the number of bytes to be mapped.

The values *hexloc1*, *hexloc2*, and *bytecount* must be from 1-8 hexadecimal digits. The leading zeros are optional.

Blanks are not allowed within a range specification, but each range must be separated from the other ranges by one or more blanks.

Options

SUM
writes STORMAP summary data to the output devices. This data includes:

VMSIZE
The size of your virtual machine in bytes.

NUCALPHA
The address of NUCALPHA, which is the starting address of the CMS nucleus.

NUCSIGMA
The address of NUCSIGMA, which is the end of executable code within the CMS nucleus and the start of the index to the message repository. The saved S- and Y-STATs are stored above the message index.

NUCOMEGA
The address of NUCOMEGA, which is the end of the CMS nucleus area.

NUCALPHA and NUCOMEGA mark the starting and ending addresses of the Named Saved System when CMS is IPLed by system name.

NUCPHI
The start of CMS nucleus storage above the 16MB line.

NUCCHI
The end of used CMS nucleus storage above the 16MB line.

TOLT16MB
Total free storage on the unallocated free storage queue below the 16MB line.

LGLT16MB
The size of the largest block of contiguous free storage on the unallocated free storage queue below the 16MB line.

TOGT16MB
Total free storage on the unallocated free storage queue above the 16MB line.

LGGT16MB
The size of the largest block of contiguous free storage on the unallocated free storage queue above the 16MB line.

TOTLUNAL
The total amount of free storage on the unallocated free storage queue both above and below the 16MB line. This is equal to TOLT16MB + TOGT16MB.

The default is SUM.

NOSUM
does not write STORMAP summary data to the output device.

HEADer
includes a header for the data items written to the output device, and separates the items with blank lines. This is the default.

NOHEADer
does not include a header, or separate the items with blank lines.

TYPe
displays the output on the console. This is the default.

NOType
does not display the output on the console. If NOTYPE is specified and the FILE option is omitted, no report is produced.

FILE
writes the output to a file called STORMAP DATA A, erasing STORMAP DATA A first if it previously existed.

APPend
Used with the FILE option, appends the output to STORMAP DATA A rather than first erasing that file.

Records written to STORMAP DATA A are fixed length fields of length 80.

ALL
produces a full STORMAP report, including the summary information and the mapping of your entire virtual machine's storage.

Specifying STORMAP (ALL is equivalent to specifying:

```
STORMAP 0.END (TYPE HEADER SUM ALLOC UNALLOC FULL PART
```

Only the following options may be specified with ALL:

- TYPE or NOTYPE
- FILE or FILE APPEND
- SUM or NOSUM
- HEADER or NOHEADER
- EXTSET
- EXTCLR
- STEM
- SUBPOOL

If ALL is specified with any other parameter, you receive an error message.

FULL
maps full pages of storage. A full page of storage is defined as a 4KB (4096 bytes) page that is either entirely allocated or entirely unallocated.

If FULL is specified without an address range or subpool name, all full pages are mapped within the constraints of the ALLOC and UNALLOC options.

Note: For more information on how the PART/FULL and ALLOC/UNALLOC sets of options interact, see Usage Note “3” on page 1042.

PART
maps partial pages of storage in terms of allocated or unallocated storage. A partial page of storage is defined as a 4KB (4096 bytes) page that is only partially in use responding to requests for free storage, but which still contains some unallocated storage.

If PART is specified without an address range or subpool name, all partial pages are mapped within the constraints of the ALLOC and UNALLOC options.

ALLOC
maps allocated blocks of storage.

If ALLOC is specified without an address range or subpool name, all allocated storage is mapped within the constraints of the PART and FULL options.

UNALloc
maps unallocated blocks of storage.

If UNALLOC is specified without an address range or subpool name, all unallocated storage is mapped within the constraints of the PART and FULL options.

SUBpool *name*
specifies the name of a subpool. Specifying a subpool name constrains all other options to act only on storage in that subpool. In this way, you can map specific regions of allocated or unallocated storage in the subpool. The subpool name can be from 1-8 characters, cannot contain blanks or nondisplayable characters, and is case-sensitive. Use the ALL option to display information about a subpool with names you cannot directly specify. For subpool names that contain nonblank, nonprintable characters, you can pass the names from a program using the extended plist.

If the SUBPOOL option is not specified, the other options (or defaults) specified are applied to ALL subpools defined in your virtual machine.

EXTSET *nnnn*
makes CMS wait until the hexadecimal external interrupt code specified by *nnnn* occurs before processing the other parameters you have specified.

EXTSET is intended to be used when you want to produce a map while the virtual machine is not at the CMS Ready message. For example, it may be in CP read, during a CP Trace of a program. If you call STORMAP with the EXTSET nnnn option before the trace starts, whenever you want the map during the trace, you can use the CP EXTERNAL command to cause the external interrupt and the subsequent STORMAP call with all the desired options already set.

Note: If the virtual machine is disabled for external interrupts at the time the CP EXTERNAL command generates the external interrupt, the previously saved STORMAP command may not be executed immediately. After completion of STORMAP, control will resume at the point where the external interrupt was generated, that is, back in CP read.

You may have as many STORMAP (or SUBPMAP) EXTSETs specified as you want, with the same code or different codes for each one. When any particular external interrupt causes STORMAP or SUBPMAP to be called, all EXTSET's saved with the specified code are called, each in turn.

It is, therefore, possible to have a setup where you have a STORMAP map going to disk, one to the console, and a SUBPMAP map going to the console as well, all when the specified external interrupt occurs.

EXTCLR *nnnn*
clears ALL saved parameter combinations for the external interrupt code *nnnn*. If the same interrupt code was in use for SUBPMAP, it will be left active for SUBPMAP. If not, a HNDEXT CLR is issued against the external interrupt code.

If STORMAP is dropped with NUCXDROP, any HNDEXT exits defined by the EXTSET option will be cleared unless active for SUBPMAP as well.

Unlike EXTSET, EXTCLR also executes other parameters if specified on the call, and displays a map. If you do not want a map in this case, specify NOSUM NOHEAD with EXTCLR.

STEM *xxxxxx*
(for invocation from REXX and EXEC2 only) specifies the character string *xxxxxx* is the prefix of a variable used to pass data to a REXX or EXEC2 program with the EXECCOMM interface. (Header information is not supplied.) The names of the individual data items requested are concatenated to this prefix and the values of the variables are set accordingly.

The STEM option is most useful when *xxxxxx* is the stem of a REXX compound variable ending with a period, like "XYZ".

If you have specified the SUM option, the items usually returned with SUM are concatenated to the prefix *xxxxxx*, resulting in REXX variables like "XYZ.NUCALPHA", if your prefix was "XYZ" ("XYZ" for EXEC2).

If you have requested storage mapping data, the line number for each line is concatenated to the prefix. (A line number for each line will be set as it would appear on the screen.) The variable *xxxxxx.0* is set to the number of lines defined. Therefore, if the mapping took 10 lines, *XYZ.0* is set to 10.

When the STEM option is used, STORMAP translates the name specified to uppercase, up to the first period, or to the end of the string.

The STORMAP command uses the "S" form of EXECCOMM. For more information on EXECCOMM, see *z/VM: REXX/VM Reference*.

Usage Notes

1. STORMAP installs itself as a nucleus extension the first time it is invoked. It may thereafter be invoked as an immediate command.
2. Calls invoked from assembler language programs by SVC 202 or CMSCALL must provide both an extended and tokenized plist. If the extended plist is not available, an error occurs.
3. Defaults

   Entering STORMAP with no options or parameters is equivalent to entering:

   ```
   STORMAP (TYPE SUM HEADER
   ```

   Similarly, the PART/FULL set of options has defaults that depend on what you specified for ALLOC/UNALLOC.

   These "basic" defaults can change when other options are specified. Table 46 on page 1042 shows what options are in effect when you specify different options.

Table 46. STORMAP Default Parameters

Parameters Specified	Equivalent to Entering
STORMAP	STORMAP (TYPE SUM HEADER
STORMAP (ALL	STORMAP (ALLOC UNALLOC PART TYPE SUM HEADER FULL
STORMAP (FILE	STORMAP (FILE NOTYPE SUM HEADER
Call STORMAP '(STEM XYZ.'	Call STORMAP '(STEM XYZ. NOTYPE SUM HEADER'
STORMAP 20000.500	STORMAP. 20000.500 (NOSUM ALLOC UNALLOC PART FULL TYPE HEADER
STORMAP (SUBPOOL USER	STORMAP (SUBPOOL USER NOSUM ALLOC UNALLOC PART FULL TYPE HEADER
STORMAP (ALLOC	STORMAP (ALLOC PART FULL TYPE NOSUM HEADER
STORMAP (UNALLOC	STORMAP (UNALLOC PART FULL TYPE NOSUM HEADER
STORMAP (PART	STORMAP (PART ALLOC UNALLOC TYPE NOSUM HEADER
STORMAP (FULL	STORMAP (FULL ALLOC UNALLOC TYPE NOSUM HEADER

Examples

```
Ready;
stormap
                                Storage Map
                                ------- ---

VMSIZE       NUCALPHA     NUCSIGMA     NUCOMEGA    NUCPHI      NUCCHI
00600000     00E00000     00F9FA08     01100000    01000000    01045358

                           Unallocated Free Storage Queue
                           ----------- ---- ------- -----

      <16MB                        >16MB
Total       Largest      Total       Largest     Total Unallocated
0042B000    00350000     00000000    00000000    0042B000
Ready;
```

Figure 47. Example of STORMAP with No Parameters or Options

```
Ready;
stormap 13000.1000 15000-20000
                                Storage Map
                                ------- ---


Address Range: 00013000 - 00013FFF

Subpool  Start    End      Bytes   Pages Key Attributes
DMSBLOKN 00010000 00013FFF 00004000    4 F0  ALLOC GLOBAL SYSTEM

Address Range: 00015000 - 00020000

Subpool Start    End      Bytes    Pages  Key Attributes
        00014000 00D3CFFF 00D29000  3369  --  UNALLOC
Ready;
```

Figure 48. Example of STORMAP with Address Ranges

Where:

Field
Description

Subpool
is the name of the subpool in which the storage resides. If no subpool name is listed, this storage is on the unallocated free storage queue.

Start
is the starting address of a block of storage in which the requested piece to be mapped resides.

End
is the ending address of a block of storage in which the requested piece of storage to be mapped resides.

A piece may be mapped by one or several blocks. Each block of storage that falls within the specified range and matches the requested attributes is displayed on a separate line with a starting and ending address.

Bytes
is the hexadecimal number of bytes from `Start` to End for the particular block of storage.

Pages
If `Bytes` is a multiple of 4096, it is displayed here as a decimal number of pages. If bytes is less than 4096, "p" is displayed, meaning it is a partially allocated page of storage.

Full and partial pages are shown as separate blocks **even** if they were obtained or released as one contiguous piece of storage. For example, if a page and a half of storage is obtained on one call, it is displayed by STORMAP as one fully allocated page of free storage followed by a half page of partially allocated storage.

Key
is the storage protection key of the page in which the block resides. Blocks of storage on the unallocated free storage queue have "--" displayed under the key field.

Attributes
are the storage attributes associated with the block of storage. The possible attributes are:

Attribute and Meaning

ALLOC
means the piece of storage is allocated. If the "pages" column is a decimal number, the piece of storage is **fully** allocated. If the "pages" column contains a lowercase "p", the piece of storage is **partially** allocated.

UNALLOC
means the piece of storage is unallocated. If the "pages" column is a decimal number, the piece of storage is **fully** unallocated. In this case, the "key" field contains "--". If the "pages" column contains a lowercase "p", the piece of storage is **partially** unallocated and the key field contains the key the page has been set to.

Fully unallocated storage is always on the unallocated free storage queue and no subpool name is displayed. Partially unallocated storage always has a subpool name displayed.

GLOBAL
means this piece of storage is in a GLOBAL subpool.

SHARED
means this piece of storage is in a SHARED subpool.

PRIVATE
means this piece of storage is in a PRIVATE subpool.

SYSTEM
means the GLOBAL subpool is SYSTEM, meaning it survives abends.

Messages and Return Codes

- DMS095E Invalid address *address* [RC=24]
- DMS389E Invalid hexadecimal number: *nnnn* [RC=24]
- DMS2512E External interrupt code *nnnn* is not set by STORMAP [RC=16]
- DMS2513E Extended plist is required. [RC=24]
- DMS2514E STEM cannot be specified outside of the REXX or EXEC2 environment. [RC=24]
- DMS2515E Invalid stem variable. [RC=24]
- DMS2516E Invalid address range: *addr1-addr2*, start greater than end. [RC=24]
- DMS2517E Error on Call to EXECCOMM, RC= *nn*. [RC=8]
- DMS2517E Error on Call to FSWRITE, RC= *nn*. [RC=20]
- DMS2518E Error, RC= *nn* from *STORMAP Initialization*. [RC=*rc*]
- DMS2518E Error, RC= *nn* from *HNDEXT SET*. [RC=16]
- DMS2519E Error detected in STORWORK savearea at address *xxxxxxxx* [RC=998]
- DMS3952E Conflicting option *option*. [RC=12]
- DMS8503E Invalid STORMAP parameter list [RC=24]

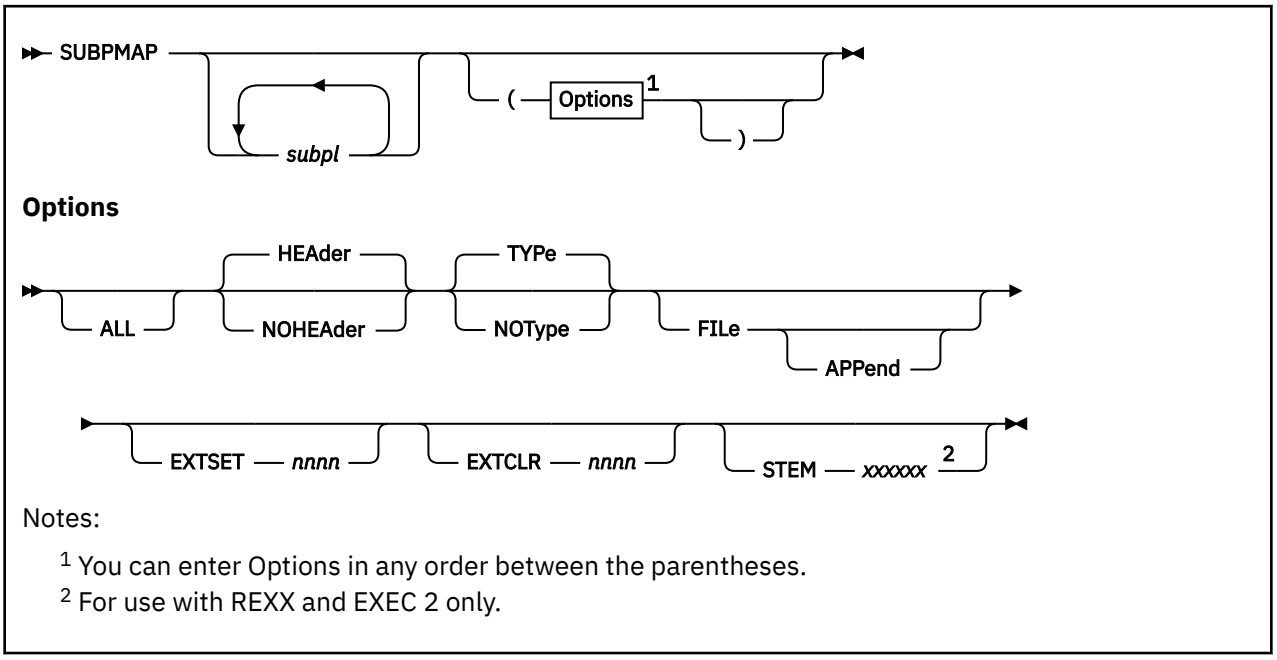
Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

Return codes:

RC	Meaning
0	Finished correctly
8	Error on EXECCOMM processing
12	Conflicting options were specified
16	Error on HNDEXT macro invocation
20	Error on FS macro file I/O
24	A not valid parameter list was specified
28	NUCXLOAD error
99	STORMAP was invoked on an unsupported level of CMS
104	Insufficient storage
998	DMSFRQ detected data corruption in the DMSFRWSW (STORWORK) control block

SUBPMAP



Notes:

[1] You can enter Options in any order between the parentheses.
[2] For use with REXX and EXEC 2 only.

Authorization

General User

Purpose

Use the SUBPMAP command to map the subpools and associated storage within your virtual machine.

SUBPMAP is used with the STORMAP and STDEBUG commands.

Operands

subpl
this operand, when repeated, specifies a list of the subpools you want to be mapped. Subpool names can be from 1-8 characters, cannot contain blanks or nondisplayable characters, and are case sensitive. Use the ALL option to display information on subpool with names you cannot directly specify. For subpool names containing nonblank, nonprintable characters, you can pass the names from a program using the extended PLIST.

Options

ALL
produces a report on all currently defined subpools. ALL may not be specified with individual subpool names.

ALL is the default if no subpool names are provided.

HEAder
includes a header for the data items written to the output device(s), and separates the data items with blank lines.

NOHEAder
does not include a header for the data items, and they are not separated by blank lines.

TYPe
displays the output on the console.

NOType
does not display the output on the console. If NOTYPE is specified and the FILE option is omitted, no report is produced.

FILe
writes the output to a file called SUBPMAP DATA A, erasing SUBPMAP DATA A first if it previously existed.

APPend
Used with the FILE option, specifies the output should be appended to SUBPMAP DATA A rather than first erasing that file.

When the FILE option is used, the records written to SUBPMAP DATA A is fixed length, of size 80.

EXTSET *nnnn*
makes CMS wait until the hexadecimal external interrupt code specified by *nnnn* occurs before processing the other parameters you have specified.

EXTSET is intended to be used when you want to produce a map while the virtual machine is not at the CMS Ready message. For example, it may be in CP read, during a CP Trace of a program. If you call SUBPMAP with the EXTSET *nnnn* option before the trace starts, whenever you want the map during the trace, you can use the CP EXTERNAL command to cause the external interrupt and the subsequent SUBPMAP call with all the desired options already set.

The virtual machine is disabled for external interrupts at the time the CP EXTERNAL command generates the external interrupt, the previously saved SUBPMAP command may not be executed immediately.) After completion of SUBPMAP, control will resume at the point where the external interrupt was generated, that is, back in CP read.

You may have as many SUBPMAP (or STORMAP) EXTSET's specified as you want, with the same code or different codes for each one. When any particular external interrupt causes SUBPMAP or STORMAP to be called, all EXTSET's saved with the specified code are called, each in turn.

It is therefore possible to have a setup where you have a SUBPMAP map going to disk, one to the console, and a STORMAP map going to the console as well, all when the specified external interrupt occurs.

EXTCLR *nnnn*
clears **all** saved parameter combinations for the external interrupt code *nnnn*. If the same interrupt code was in use for STORMAP, it is left active for STORMAP. If not, a HNDEXT CLR is issued against the external interrupt code.

If SUBPMAP is dropped with NUCXDROP, any HNDEXT exits defined by the EXTSET option will be cleared unless active for STORMAP as well.

Unlike EXTSET, EXTCLR will also execute other parameters if specified on the call, and will display a map.

STEM *xxxxxx*
(for invocation from REXX and EXEC2 only) specifies the character string *xxxxxx* is the name of a prefix used to pass data to a REXX or EXEC2 program with the EXECCOMM interface. (Header information is not supplied.)

The STEM option is most useful when the name is a REXX compound variable ending with a period, for example, "XYZ.".

The line number for each line (as it would appear on the screen) is concatenated to the prefix, so REXX variables like *XYZ.1*, *XYZ.2*, and so forth, can be set. (For EXEC2, the variables would be XYZ1, XYZ2, so forth.) Each line of screen information is put into the corresponding variable. The variable *name.0* is set to the number of lines defined. Therefore, if the mapping took 10 lines, *XYZ.0* is set to 10.

When the STEM option is used, SUBPMAP translates the name specified to uppercase, up to the first period, or to the end of the string.

SUBPMAP uses the "S" form of EXECCOMM. For more information on EXECCOMM, see *z/VM: REXX/VM Reference*.

Usage Notes

1. SUBPMAP installs itself as a nucleus extension the first time it is invoked. It may thereafter be invoked as an immediate command.
2. Calls invoked from assembler language programs by SVC 202 or CMSCALL must provide both an extended and tokenized plist. If the extended plist is not available, an error occurs.

Examples

```
Ready;
subpmap
                                  Subpool Map
                                  ------- ---

Subpool   Key   Anchor        Full      Part  Attributes
DMSINTSP  F0    01014B8C       547         0  GLOBAL  SYSTEM
DMSBLOKN  F0    01014BB4       160        15  GLOBAL  SYSTEM
NUCLEUS   F0    01014C04        11         6  GLOBAL  SYSTEM
USER      E0    01014C2C         0         0  GLOBAL
DMSBLOKU  E0    01014C54         0         1  GLOBAL
DMSUSER   E0    01014CA4         0         0  GLOBAL
DMSLDSET  F0    01014CF4         0         0  GLOBAL
DMSBLOKS  F0    01014BDC         0         0  GLOBAL  SYSTEM
DMSUSRM   E0    01014CCC         1         2  GLOBAL
PRIV1     E0    01014E5C         0         1  PRIVATE
SHAR1     E0    01014E84         0         1  SHARED
PRIV1     E0    01014D44         0         1  PRIVATE
DMSDCSUS  F0    01014D94         0         0  GLOBAL
DMSDCSYS  F0    01014D6C         0         0  GLOBAL  SYSTEM
Ready;
```

Figure 49. Example of SUBPMAP with No Parameters or Options

```
Ready;
subpmap USER NUCLEUS
                                  Subpool Map
                                  ------- ---

Subpool   Key   Anchor        Full      Part  Attributes
NUCLEUS   F0    01014C04        11         6  GLOBAL  SYSTEM
USER      E0    01014C2C         0         0  GLOBAL
Ready;
```

Figure 50. Example of SUBPMAP with Subpools Specified

Where:

Field
Description

Subpool
is the name of the subpool.

Key
is the storage protection key of the page the block resides within.

Anchor
is the address of the subpool descriptor block for the subpool being displayed.

Full
is the decimal number of full pages of allocated storage on the specified subpool.

Part
is the decimal number of partial pages of allocated storage on the specified subpool.

Attributes
are the storage attributes associated with the block of storage. The possible attributes are:

Attribute	Meaning
GLOBAL	the subpool that this piece of storage is associated with is GLOBAL in scope.
SHARED	the subpool that this piece of storage is associated with is SHARED in scope.
PRIVATE	the subpool that this piece of storage is associated with is PRIVATE in scope.
SYSTEM	the GLOBAL subpool is SYSTEM, meaning it survives abends.

Messages and Return Codes

- DMS389E Invalid hexadecimal number: *nnnn* [RC=24]
- DMS2512E External interrupt code *nnnn* is not set by SUBPMAP [RC=16]
- DMS2513E Extended plist is required. [RC=24]
- DMS2514E STEM cannot be specified outside of the REXX or EXEC2 environment. [RC=24]
- DMS2515E Invalid stem variable. [RC=24]
- DMS2517E Error on Call to EXECCOMM, RC= *nn*. [RC=8]
- DMS2518E Error on Call to FSWRITE, RC= *nn*. [RC=20]
- DMS2518E Error, RC= *nn* from *SUBPMAP Initialization*. [RC=*rc*]
- DMS2518E Error, RC= *nn* from *HNDEXT SET*. [RC=16]
- DMS2519E Error detected in STORWORK savearea at address *xxxxxxxx* [RC=998]
- DMS3952E Conflicting option *option*. [RC=12]
- DMS8503E Invalid SUBPMAP parameter list [RC=24]

Return codes:

RC	Meaning
0	Finished correctly
8	Error on EXECCOMM processing
10	One or more subpool names specified were not found. If multiple names were supplied, the ones that were found will be mapped.
12	Conflicting options were specified
16	Error on HNDEXT macro invocation
20	Error on FS Macro file I/O
24	A not valid parameter list was specified
28	NUCXLOAD error

99
SUBPMAP was invoked on an unsupported level of CMS

104
Insufficient storage

998
DMSFRQ detected data corruption in the DMSFRWSW (STORWORK) control block

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

SVCTRACE

Authorization

General User

Purpose

Use the SVCTRACE command to trace and record information about supervisor calls occurring in your virtual machine.

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 SVCs 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 *dddddddd*
       = h h h h h h h h *dddddddd*
```

Where:

h
specifies the contents of a general register in hexadecimal format

d
specifies 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
specifies the hexadecimal contents of a general register

d
specifies the EBCDIC translation of the contents of a general register.

The only general registers 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 *dddddddd*
       = h h h h h h h h *dddddddd*
```

Where:

h
specifies the hexadecimal contents of a general register

d
specifies 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
specifies the hexadecimal contents of a floating-point register

g
specifies 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
specifies the hexadecimal contents of a floating-point register

g
specifies 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 *dddddddd*
      = h h h h h h h h *dddddddd*
```

Where:

h
represents a word of hexadecimal data

d
specifies 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 (*).

The Table 47 on page 1054 summarizes the types of SVC trace output.

Table 47. 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.

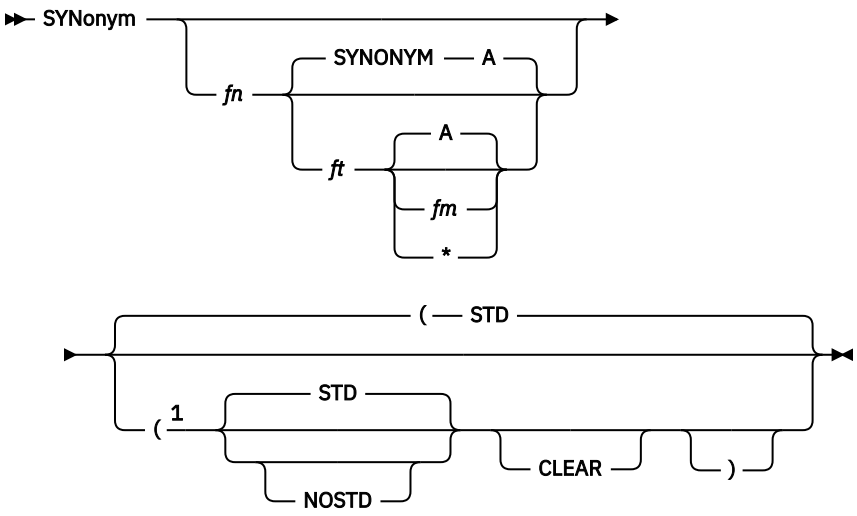
Messages and Return Codes

- DMS014E Invalid function *function* [RC=24]
- DMS047E No function specified [RC=24]
- DMS104S Error *nn* reading file *fn ft fm* from disk or directory [RC=31|55|70|76|99|100]
- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS257T Internal system error at address *addr* (offset *offset*)
- DMS1144E Implicit rollback occurred for work unit *workunitid* [RC=31]
- DMS1252T Rollback unsuccessful for file pool *filepoolid*

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411
Errors in the Shared File System	“File Pool Server Messages” on page 1414
Errors in using a file	“File Error Messages” on page 1418

SYNONYM



Notes:

[1] You can enter Options in any order between the parentheses.

Authorization

General User

Purpose

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** section called "The User Synonym Table".

The names you define can be used either instead of or 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.

Operands

fn
is the file name of the file containing your synonyms table.

ft
is the file type of the file containing your synonyms table. SYNONYM is the default file type.

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 standard CMS abbreviations are accepted. This is the default.

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, 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. An exec procedure can be invoked by its synonym if the implied exec (IMPEX) function is on. However, within a CMS EXEC or EXEC2 procedure, only the file name of the exec being invoked can be used. A synonym is not recognized within a CMS EXEC or EXEC2 procedure because the synonym tables are not searched during processing of these procedures. Synonym tables can be checked during REXX exec processing, so synonyms can be used to invoke another exec.
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. For more information, see the MYSYN example below.

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 because no count is specified. If MYSYN SYNONYM contains:

```
ACCESS GETDISK 3
```

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 you can invoke the exec procedure by specifying the truncation TD.

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 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 USENAME 3
```

USENAME is a synonym for FILEDEF, and acceptable truncations of USENAME are: USE, USEN, USENA, USENAM, and USENAME. 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 48 on page 1058 shows the forms of the system command and user synonyms available for the various combinations of the SET ABBREV and SYNONYM commands.

Table 48. System and User-Defined Truncations

Options	Acceptable Command Forms	Comments
SET ABBREV ON SYNONYM USERTAB (STD	FI FIL ⋮ FILEDEF USE USEN ⋮ USENAME	The ABBREV ON option of the SET command and the STD option of the SYNONYM command make the system table available. The user synonym, USENAME, is available because the synonym table (USERTAB) is specified on the SYNONYM command. The truncations for USENAME are available because SET ABBREV ON was specified with the USERTAB also available.
SET ABBREV OFF SYNONYM USERTAB (STD	FILEDEF USENAME	The user-defined synonym, USENAME, is permitted because the user synonym table (USERTAB) is specified on the SYNONYM command. No system or user truncations are permitted.
SET ABBREV ON SYNONYM USERTAB (NOSTD	FILEDEF USE USEN ⋮ USENAME	The system synonym table is unavailable because the NOSTD option is specified on the SYNONYM command. The user synonym, USENAME, is available because the user synonym table (USERTAB) is specified on the SYNONYM command and the truncations of USENAME are permitted because SET ABBREV ON is specified with USERTAB also available.

Table 48. System and User-Defined Truncations (continued)

Options	Acceptable Command Forms	Comments
SET ABBREV OFF SYNONYM USERTAB (NOSTD	FILEDEF USENAME	The system synonym table is made unavailable either by the SET ABBREV OFF command or by the SYN (NOSTD command. The synonym, USENAME, is permitted because the user-defined synonym table (USERTAB) is specified on the SYNONYM command. The truncations for USENAME are not permitted because the SET ABBREV OFF option is in effect.
SET ABBREV ON SYNONYM (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.
SET ABBREV OFF SYNONYM (CLEAR STD SET ABBREV ON SYNONYM (CLEAR NOSTD SET ABBREV OFF SYNONYM (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.

Note: 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     USER       SHORTEST
command    synonym    form (if any)
   .          .          .
   .          .          .
   .          .          .
```

This response is the same as the response to the command QUERY SYNONYM ALL.

```
DMS711I  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.

```
DMS712I  No synonyms (DMSINA not in nucleus)
```

The system routine which handles SYNONYM command processing is not in the system.

Messages and Return Codes

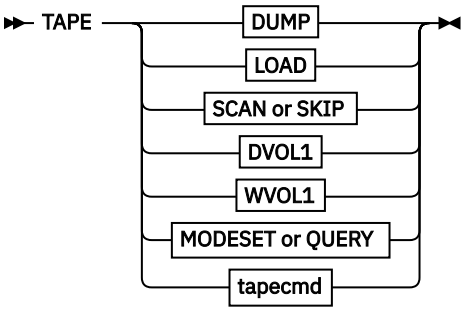
- DMS002E File [*fn* [*ft* [*fm*]]] not found [RC=28]
- DMS003E Invalid option: *option* [RC=24]
- DMS007E File *fn ft fm* [is] not fixed, 80-character records [RC=32]
- DMS056E File *fn ft* [*fm*] contains invalid record formats[RC=32]
- DMS066E *option1* and *option2* are conflicting options [RC=24]
- DMS104S Error *nn* reading file *fn ft fm* from disk or directory [RC=31|55|70|76|99|100]
- DMS257T Internal system error at address *addr* (offset *offset*)

- DMS1144E Implicit rollback occurred for work unit *workunitid* [RC=31]
- DMS1229E *fn ft fm* is empty [RC=88]
- DMS1252T Rollback unsuccessful for file pool *filepoolid*

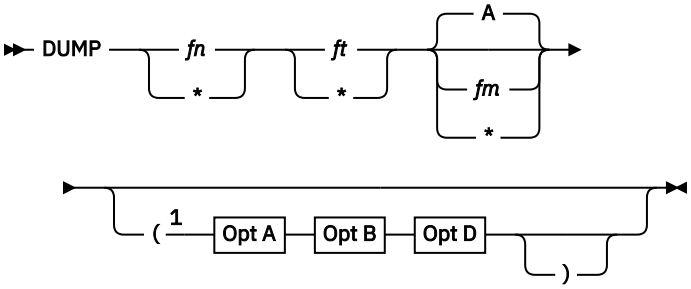
Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411
Errors in using a file	“File Error Messages” on page 1418

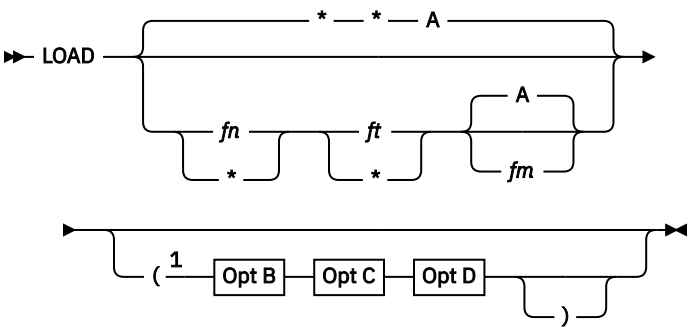
TAPE



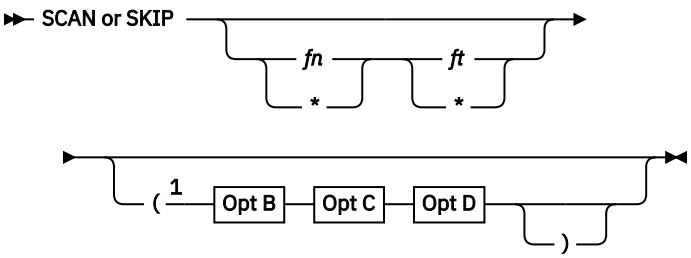
DUMP



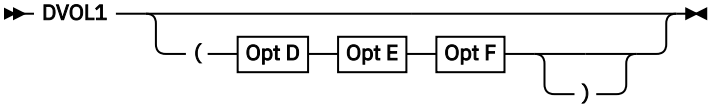
LOAD



SCAN or SKIP



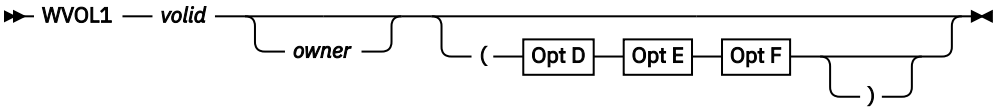
DVOL1



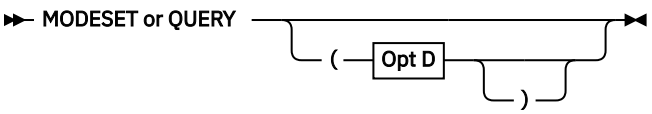
Notes:

[1] You can enter Options in any order between the parentheses.

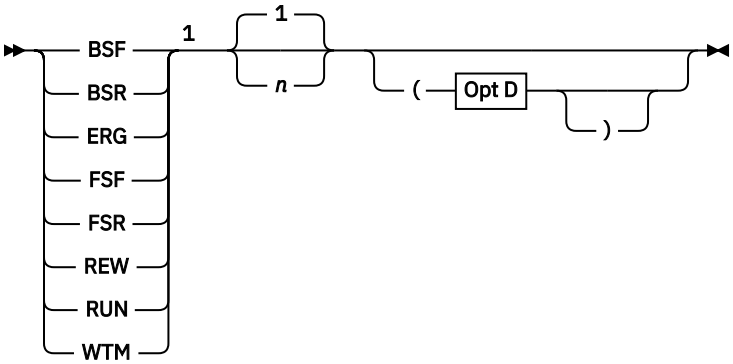
WVOL1



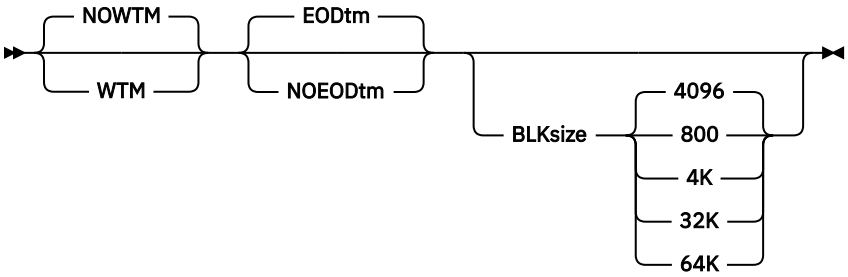
MODESET or QUERY



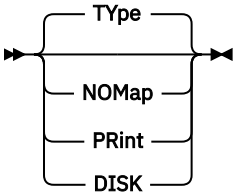
Tapecmd



Opt A



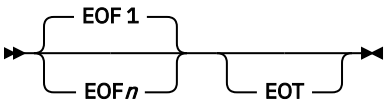
Opt B



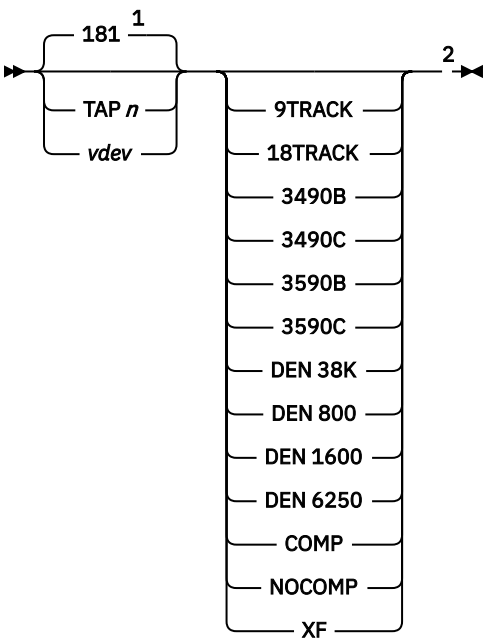
Notes:

[1] ERG, REW, and RUN do not require a numeric value.

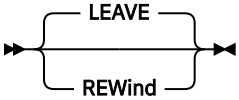
Opt C



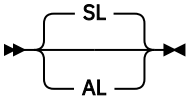
Opt D



Opt E



Opt F



Notes:

[1] You can also specify TAP1. TAP1 is a synonym for 181.

[2] In general, only one recording format should be used. However, for compatibility reasons, it is allowed to specify 18TRACK with DEN38K, or 9TRACK with DEN 800, DEN 1600, or DEN 6250.

Authorization

General User

Purpose

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 multivolume files. Files to be dumped can contain either fixed-length or variable-length records.

To load files from VM Product and Service tapes, see “VMFPLC2” on page 1131.

Operands

DUMP

dumps one or more CMS files to tape. You must specify *fn* and *ft*; *fm* is optional. If you do not specify the file mode, the default is A.

If *fn ft* are fully specified (such as MY FILE), extensions of your A disk or directory will not be picked up.

If you specify file name or file type, or both, as an asterisk (*), all files that satisfy the other file identifier(s) are dumped, including files on extensions of the file mode specified (or the default of A). If you specify file mode as an asterisk (*), all file modes are searched for the specified file(s).

LOAD
reads tape files to a disk or directory. If you specify a file identifier with no asterisks, only that single specified file is loaded. You must be positioned after a volume label (if one exists) to use the LOAD option. If you specify the EOF *n* option without a file identifier, *n* tape files are loaded.

Note: The *tape file* consists of all the CMS files between two tape marks. If you specify *fn* or *ft* as an asterisk, all files within the range of EOF *n* that satisfy the other file identifier are loaded. (You must specify both identifiers.)

The files are written to the specified file mode. If you do not specify a file mode, the default is A. If you also specify a file mode number, only files with that file mode number are loaded. If you do not specify file name, file type, or file mode, all files up to EOF or end-of-tape are loaded to your disk or directory accessed as A.

SCAN
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 you specify *fn* and *ft*, scanning stops upon encountering that file, with the tape positioned to load the specified file.

You must be positioned after a volume label (if one exists) to use SCAN.

DVOL1
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
writes a VOL1 label on a tape. All fields are set to the same values that are set when a VOL1 label is written by the IBM-supplied IEHINITT utility program. For more information, see *Magnetic Tape Labels and File Structure for MVS/DFP*.

volid
The volume ID is set to the 1-6 character volume ID specified on the command.

owner
If *owner* is specified with WVOL1, the owner name 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. For more information, see Options E.

Note: A valid *volid*s can be uppercase alphabetic, numeric, or the following characters:

For AL tapes: " ; : < = > ? - . + / , * & ! ( ) '
For SL tapes: ¢ $ # - / @

You may want to ensure any tape mounting programs used by your tape library support special characters in tape labels.

SKIP
positions the tape at a specified point and lists only the identifier of the file it skips. Skipping occurs over *n* tape marks, as specified by the option EOF *n* (the default is 1 tape mark). However, if you specify *fn* and *ft* with SKIP, skipping stops after encountering that file, with the tape positioned immediately following the file.

QUERY
displays information about a tape device, particularly which recording formats the device is capable of writing.

In response to the QUERY option, CMS issues the following message for each of the recording formats which the device is capable of writing:

```
DMS217I  Device vdev can write recording format (option)
```

MODESET
sets default recording options. All subcommands set the default recording format options (just like MODESET, but the others do something else as well; MODESET does nothing else). For more information on setting default recording format options, see Usage Note "9" on page 1069, and *z/VM: CMS User's Guide*.

Note: Before you use the MODESET option, you should rewind your tape to ensure the tape is at load point. Entering the command TAPE REW, positions the tape at load point.

tapcmd n
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 *n* tape marks
BSR	Backspace *n* tape records
ERG	Erase a defective section of the tape
FSF	Forward-space *n* tape marks
FSR	Forward-space *n* tape records
REW	Rewind tape to load point
RUN	Rewind tape and unload
WTM	Write *n* tape marks

Options

If conflicting options are specified, an error message will be issued.

Options A

WTM
writes a tape mark after each file dumped.

NOWTM
does not write a tape mark after each file dumped. The default is NOWTM.

EODtm
writes two tape marks after the last file dumped. (Two tape marks indicate the end-of-tape (EOT).) It then backspaces over one or two tape marks to position the tape in case more files are to be dumped. The default is EODTM.

NOEODtm
writes no additional tape marks at the end of the last file dumped.

Attention: If two tape marks do not follow the last file on a tape, some tapes like the 3480 may give unpredictable results. If you specify NOEODTM, you should ensure there are at least two tape marks at the end of your *final* dump by either using the default EODTM or issuing `TAPE WTM 2` after your final dump.

Table 49 on page 1066 shows the results of specifying various combinations of EODTM, NOEODTM, WTM, and NOWTM:

Table 49. Results of EODTM, NOEODTM, WTM and NOWTM Option Combinations

Options Specified	Tape Marks Between Files	Tape Marks at End of Files	Tape is Positioned
WTM WTM EODTM	1	2	Between last 2 tape marks
WTM NOEODTM	1	1	After last tape mark
NOWTM NOWTM EODTM EODTM	0	2	Before last 2 tape marks
NOWTM NOEODTM NOEODTM	0	0	At end of files

BLKsize
specifies the approximate size of the tape block to which the files are dumped. The larger the block size, the more data will fit on a tape. The data capacity of a tape depends on the BLKSIZE option used:

BLKsize
Data Capacity

800
up to 800 bytes, plus control information

4K (4096)
up to 4096 bytes, plus control information. This is the default

32K
up to 32,767 (32K-1) bytes, plus control information (maximum supported for the 9346 tape unit)

64K
up to 65,535 (64K-1) bytes, plus control information

TRANsfer BUFF
TRANsfer IMMED
This option does nothing. It exists only for compatibility with previous releases of VM and may be removed in a future release.

Options B

Type
displays a list of files dumped, loaded, scanned, or skipped at the terminal. This is the default.

Note: This option replaces the TERM option. TERM is still functional, but is no longer supported.

NOMap
suppresses all MAP output. The list of files processed is not spooled to any output device (console, printer, or disk), and no TAPE MAP file is created.

Note: This option replaces the NOPRint option. NOPRint is still functional, but is no longer supported.

PRint
spools the list of files dumped, loaded, scanned, or skipped to the user's virtual printer. For more information, see Usage Note “16” on page 1069.

DISK
creates a CMS file called TAPE MAP A5 containing the list of files dumped, loaded, scanned, or skipped.

Options C

EOF *n*
reads the tape through a maximum of *n* tape marks. If the EOF option is not specified, the default is EOF 1. If EOF *n* is specified, you must specify a number for *n*, or you will receive error DMS393E.

EOT
reads the tape until an end-of-tape indication (two tape marks) is recognized. You may want to specify EOT after EOF *n* to prevent going past the end of the tape if *n* is greater than the actual number of tape marks.

Options D

TAP*n*
vdev
specifies the device name (TAP*n*), or alternatively the virtual device number (*vdev*) of the tape device on which the command is to operate. These names and corresponding virtual device numbers are valid. For more information on device names and virtual device numbers for tape devices, see *z/VM: CMS User's Guide*.

Device Name	Device Number	Device Name	Device Number
TAP0	0180	TAP8	0288
TAP1	0181	TAP9	0289
TAP2	0182	TAPA	028A
TAP3	0183	TAPB	028B
TAP4	0184	TAPC	028C
TAP5	0185	TAPD	028D
TAP6	0186	TAPE	028E
TAP7	0187	TAPF	028F

You may omit the leading zero on the device numbers. The default is TAP1.

COMP
Specifies "any compacted recording format." CMS selects a compacted recording format the device is capable of writing (if there is one). If there is a choice of formats, CMS will choose the one which will fit the greatest amount of data on the tape. If you require a particular compacted recording format, use the 3590C, 3490C or XF option.

NOCOMP
Specifies "any uncompacted recording format." CMS selects an uncompacted recording format the device is capable of writing. If there is a choice of formats, CMS will choose the one which will fit the greatest amount of data on the tape. If you require a particular uncompacted recording format, use the 3590B, 3490B, 18TRACK, DEN 6250, DEN 1600, or DEN 800 option.

3590B
specifies 3590 Basic recording format.

3590C
specifies 3590 Compacted recording format.

3490B
specifies 3490 Basic recording format.

3490C
specifies 3490 Compacted recording format.

XF
specifies 3480 Compacted recording format.

DEN 38K
specifies the 3480 Basic recording format.

DEN 38K is equivalent to 18TRACK, which is the preferred option.

DEN 800
specifies NRZI (800 BPI) recording format.

DEN 1600
specifies PE (1600 BPI) recording format.

DEN 6250
specifies GCR (6250 BPI) recording format.

18TRACK
specifies 3480 Basic recording format.

9TRACK
specifies "any 9 track recording format." CMS selects a 9 track recording format for you that the device is capable of writing (if there is one). If you require a particular 9 track recording format, use the DEN 800, DEN 1600, or DEN 6250 option.

Options E

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.

If an error occurs while reading or writing the VOL1 label, the normal positioning implied by these options will not be attempted. Because the tape position is undetermined after an error, the user must enter a separate TAPE REWind command to reposition the tape to the load point.

Options F

SL
specifies an IBM standard VOL1 label is to be displayed or written. This is the default.

AL
specifies an ANSI standard VOL1 label is to be displayed or written.

Usage Notes

1. Entering the TAPE command for a tape drive that is not ready may delay the user without any error indication. Therefore, ensure a tape drive is ready before entering the TAPE command.
2. If using a language other than American English (AMENG), you must specify the full word "REWIND" for the REWIND option. You cannot use the abbreviation.
3. A corrupt file (for example, one having an incorrect FST) may not be detected.
4. If a tape label is read while executing either a LOAD or SCAN option, message DMS057E will be issued because a tape label is not a valid record format. Either use TAPE FSR to skip over the tape label when at the load point or ignore the error message and reissue the LOAD or SCAN command to process the data records after the tape label.
5. If the tape is not correctly positioned (at the beginning of a file) at the start of an operation, this may not be detected and a corrupt file will result.
6. The first time TAPE is issued, module DMSP2C is loaded as a nucleus extension. It remains there until NUCXDROP, IPL, or LOGOFF is issued.
7. If TAPE is defined as a synonym of the VMFPLC2 command (or the VMFPLC2 command is defined as a synonym of TAPE), you cannot use the synonym to call the function from within an exec. You can 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; however, TAP is a valid synonym.

8. If there is not enough space on a file mode to hold the files contained in the tape file being loaded, load operations will terminate. To prevent this, when you dump the files, separate CMS (logical) files by tape marks, then space forward to the appropriate CMS file. For more information on tape marks on TAPE DUMP tapes, see *z/VM: CMS User's Guide*.
9. Whenever you invoke the TAPE command, no matter what subcommand you use, the command sets a "default recording format option". This controls all future writes to the device, by any method, which do not specify a particular recording format. For example:

   ```
   TAPE REW (TAP2 DEN 6250
   TAPE DUMP MYFILE SCRIPT A (TAP2
   ```

   causes MYFILE SCRIPT A to be dumped in GCR format, if device TAP2 is capable of writing GCR.

   Furthermore,

   ```
   FILEDEF INMOVE DISK MYFILE SCRIPT A
   FILEDEF OUTMOVE TAP2
   TAPE REW (TAP2 DEN 6250
   MOVEFILE INMOVE OUTMOVE
   ```

   also causes the file to be dumped in GCR format, if device TAP2 is capable of writing GCR.

   But CMS only uses the default recording format options as a last choice. For example,

   ```
   FILEDEF INMOVE DISK MYFILE SCRIPT A
   FILEDEF OUTMOVE TAP2 (DEN 1600
   TAPE REW (TAP2 DEN 6250
   MOVEFILE INMOVE OUTMOVE
   ```

   writes in PE format because the DEN 1600 on the FILEDEF command takes precedence over the DEN 6250 on the TAPE command.

   Furthermore, if CMS tries to use the default recording format option and finds the device is not capable of writing that format, it resets the default recording format option to "none" (as if you had just IPLed CMS) and just chooses the recording format according to a built in default.

   Note: This could happen if you detached a tape device and then attached one with different capabilities with the same device number. A tape operation will never fail because CMS uses the default recording format option.

   You can use the TAPE MODESET command if all you want to do is set the default recording format option.
10. In general, you should use only one recording format option. However, for compatibility with previous levels of VM, it is allowed to specify 18TRACK together with DEN 38K or 9TRACK together with DEN 800, DEN 1600, or DEN 6250.
11. For more information on the tape recording formats and how CMS deals with them, see *z/VM: CMS User's Guide*.
12. For more information about tape file handling, see *z/VM: CMS Application Development Guide for Assembler*.
13. You cannot load a file if a file precedes it on the tape that takes up more space than remains on the disk you are loading to. (This is because TAPE temporarily loads any files preceding the desired file onto your disk.) You will receive error message DMS105S, but depending on how you issued the command, the file name referred to by the message may not be correct. If this error occurs, use the SCAN option to position the tape after the large file.
14. If you try to load an empty file into a minidisk or into a directory in a file pool that does not support empty files, no file is created and you receive error message DMS636W. Empty files can only be loaded to SFS directories that support empty files.
15. For more information on the TAPE command, see *z/VM: CMS User's Guide*.
16. If the PRINT option is specified and the LPP option on the CP SPOOL command is 0 (LPP OFF), the number of lines per page for the output spooled to the printer will be 60. If the LPP value is greater than 0, the number of lines per page for the output spooled to the printer will be equal to the LPP

value minus the number of header lines. For more information, see *z/VM: CP Commands and Utilities Reference*.

17. If the tape is under the control of a Tape Library Dataserver machine, and the DFSMS/VM Removable Media Services (RMS) FSMPPSI CSLLIB is available to CMS, the RUN function calls the RMS FSMRMDMT (Demount) CSL routine to have the Dataserver unmount the tape.

Responses

If the TYPE option is in effect, the following MAP 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
.  .  .
.  .  .
.  .  .
```

Messages and Return Codes

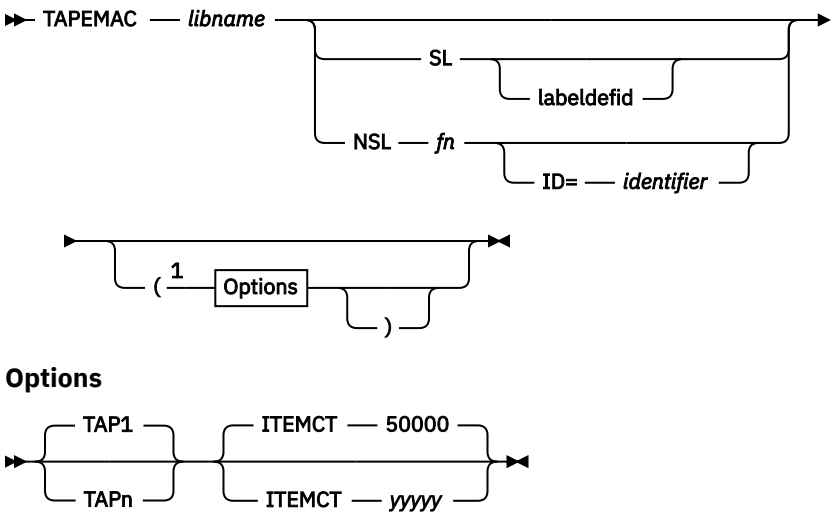
- DMS002E File *fn ft fm* not found [RC=28]
- DMS003E Invalid option: *option* [RC=24]
- DMS003E Invalid option: *option* with function *function* [RC=24]
- DMS010E Premature EOF on file *fn ft* [RC=40]
- DMS014E Invalid function *function* [RC=24]
- DMS023E No filetype specified [RC=24]
- DMS027E Invalid device *vdev* [RC=24]
- DMS029E Invalid parameter *parameter* in the option *option* field [RC=24]
- DMS037E Filemode *filemode* is accessed as read/only [RC=36]
- DMS042E No fileid(s) specified [RC=24]
- DMS043E TAPn (*vdev*) is file protected [RC=36]
- DMS047E No function specified [RC=24]
- DMS048E Invalid filemode mode [RC=24]
- DMS057E Invalid record format [RC=32]
- DMS058E End-of-file or end-of-tape [RC=40|32]
- DMS065E option *option* specified twice [RC=24]
- DMS066E *option1* and *option2* are conflicting options [RC=24]
- DMS069E Filemode *filemode* not accessed [RC=36]
- DMS104S Error *rc* reading file *fn ft fm* from disk or directory [RC=31|55|70|76|99|100]

- DMS105S Error *rc* writing file *fn ft fm* on disk or directory [RC=24|31|55|70|76|99|100]
- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS110S Error reading TAPn(*vdev*) [RC=100]
- DMS111S Error writing TAPn(*vdev*) [RC=100]
- DMS113S TAPn(*vdev*) not attached [RC=100]
- DMS115S Device *name* cannot write the *format* recording format [RC=88]
- DMS115S Device *name* cannot write any {9TRACK|compacted} recording formats [RC=88]
- DMS173W Empty output file *fn ft fm* not created [RC=0]
- DMS217I Device *vdev* can write *recording format (option)*
- DMS335W TAPn(*vdev*) has been rewound and unloaded by operator. Requested tape function may not have been executed. [RC=4]
- DMS393E Missing valuetype for option *option* [RC=24]
- DMS431E TAP*n*(*vdev*) VOL1 label missing
- DMS613E TAPE/VMFPLC2 must be invoked as a nucleus extension [RC=40]
- DMS636W File *fn ft fm* is empty; minidisk does not support empty files [RC=0, processing will continue]
- DMS671E Error loading file *fn ft fm*; rc=*rc* from RENAME [RC=31|55|70|76|99|100]
- DMS671E Error loading file *fn ft fm*; rc=*rc* from COPYFILE [RC=31|55|70|76|99|100]
- DMS925E I/O error on screen [RC=100]
- DMS1128E TAPn (*vdev*) user requested [AL|SL] standard, but [SL|AL] found [RC=32]
- DMS1262S Error *rc* opening file *fn ft fm* [RC=31|55|70|76|99|100]
- DMS1262S Error *rc* closing file *fn ft fm* [RC=31|55|70|76|99|100]
- DMS2139I VDEV *vdev* SENSE gives ERA/RAC=*rc*; cartridge may not be valid for I/O
- DMS2147W Library Dataserver DEMOUNT error CSLRC= *nnn*, CSLRS= *nnnnn*, FCTRC= *nnn*, FCTRS= *nnnnn*, a direct REWIND and UNLOAD will now be attempted

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411
Errors in the Shared File System	“File Pool Server Messages” on page 1414

TAPEMAC



Notes:

[1] You can enter Options in any order between the parentheses.

Authorization

General User

Purpose

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.

Operands

libname
specifies the file name of the first, or only, CMS MACLIB to be created on the disk or directory accessed as A. If *libname* MACLIB already exists on the disk or directory accessed as A, the old one is erased; no warning message is issued.

SL
means 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 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 1-8 character identifier specified as the *filename* on the previously issued LABELDEF command.

fn
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 used for NSL routines with the TAPEMAC command must be created so they start at an address above X'21000'.

This prevents the NSL modules from overlaying the command. For more information on how to write routines to process nonstandard labels, see "Tape Labels in CMS" in *z/VM: CMS Application Development Guide for Assembler*.

ID=*identifier*
specifies a 1-8 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. For more information on communicating with routines that process nonstandard labels, see *z/VM: CMS Application Development Guide for Assembler*.

Options

TAP*n*
specifies the device name for the virtual tape device from which to read.

TAP*n* is one of the following device names for a virtual tape device: TAP0, TAP1, TAP2, TAP3, TAP4, TAP5, TAP6, TAP7, TAP8, TAP9, TAPA, TAPB, TAPC, TAPD, TAPE, TAPF. For more information on device names and virtual device numbers for tape devices, see *z/VM: CMS User's Guide*.

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 *libname* 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 (*libname*) 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 can 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 with *labeldefid* 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 nsl nsl3
```

CMS uses your own routine NSL3 to process tape labels before loading MAC10.

Responses

The TAPEMAC command displays the message:

```
LOADING libname MACLIB
```

for each macro library created.

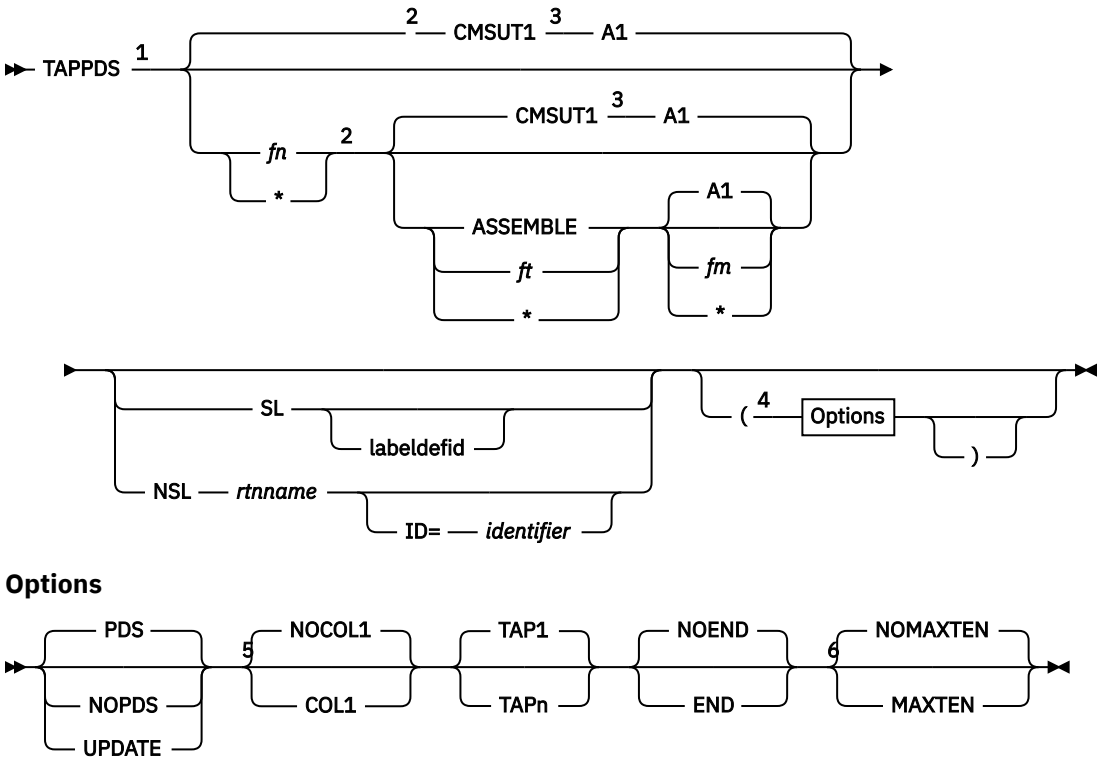
Messages and Return Codes

- DMS001E No filename specified [RC=24]
- DMS003E Invalid option: *option* [RC=24]
- DMS027E Invalid device *devtype* [for SYSaaa] [RC=24]
- DMS057E Invalid record format [RC=32]
- DMS069E Filemode *mode* not accessed [RC=36]
- DMS070E Invalid parameter *parameter* [RC=24]
- DMS105S Error *nn* writing file *fn ft fm* on disk or directory [RC=100]
- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS110S Error reading tap*n*(*vdev*) [RC=100]
- DMS138S Error *nn* erasing *fn ft* before loading tape [RC=100]
- DMS139S Tape file exceeds 9 CMS MACLIBs [RC=104]
- DMS335W Tap*n* [(*vdev*)] has been manually rewound and unloaded. Requested tape function may not have been executed. [RC=4]
- DMS420E NSL exit filename missing or invalid [RC=24]
- DMS1144E Implicit rollback occurred for work unit *workunitid* [RC=31]
- DMS1252T Rollback unsuccessful for file pool *filepoolid*

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in the Shared File System	“File Pool Server Messages” on page 1414
Errors in using a file	“File Error Messages” on page 1418

TAPPDS



Options

Notes:

[1] The *fn*, *ft*, and *fm* cannot be omitted if SL or NSL is specified.
[2] If NOPDS or UPDATE is specified and *fn* is either not specified or specified as an asterisk, the default file name is TAPPDS.
[3] The default file types are CMSUT1 (for PDS or NOPDS) and ASSEMBLE (for UPDATE).
[4] You can enter Options in any order between the parentheses.
[5] The COL1 should be used with the UPDATE option.
[6] The MAXTEN is valid only with the PDS option.

Authorization

General User

Purpose

Use the TAPPDS command to create CMS files from tapes 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 can be blocked or deblocked 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.

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 an asterisk (*).

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 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 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 1-8 character specified as the file name on the LABELDEF command that was previously issued.

rtname
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 used for NSL routines with the TAPPDS command must be created so they start at an address above X'21000'. This prevents the MODULE files from overlaying the command. For more information, on writing routines to process nonstandard labels, see "Tape Labels in CMS" in *z/VM: CMS Application Development Guide for Assembler*.

ID=*identifier*
specifies a 1-8 character identifier to a user-written NSL routine. You can 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. For more information on communication with routines that process nonstandard labels, see "Tape Labels in CMS" in *z/VM: CMS Application Development Guide for Assembler*.

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 can 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 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 IEBPTPCH service program if this option is specified. This is the default.

NOPDS
indicates the contents of the tape will be placed in one CMS file.

UPDATE
indicates 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. For more information, see Usage Note “2” on page 1077.

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 IEBPTPCH service program. This is the default.

TAP*n*
specifies the device name for the virtual tape device from which to read.

TAP*n* is one of the following device names for a virtual tape device: TAP0, TAP1, TAP2, TAP3, TAP4, TAP5, TAP6, TAP7, TAP8, TAP9, TAPA, TAPB, TAPC, TAPD, TAPE, TAPF. For more information on device names and virtual device numbers for tape devices, see *z/VM: CMS User's Guide*. The default tape device is TAP1.

END
considers an END statement (characters ‘END ’ plus one blank space in columns 2-5) a delimiter for the current member.

NOEND
specifies END statements are not to be treated as member delimiters, but are to be processed as text. This is the default.

MAXTEN
reads up to ten members. This is valid only if the PDS option is selected.

NOMAXTEN
reads any number of members. This is the default.

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, TAPPDS 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). To position the tape to the *n*th file, where *n* is the position of the file on the tape, the user needs to skip over physical tape files (3n-3 physical tape files if positioning to the header labels, 3n-2 if positioning to the data file).
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
   ```

   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

```
DMS703I  File fn ft [fm] copied
```

The named file is copied to a disk or directory.

```
DMS707I  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.

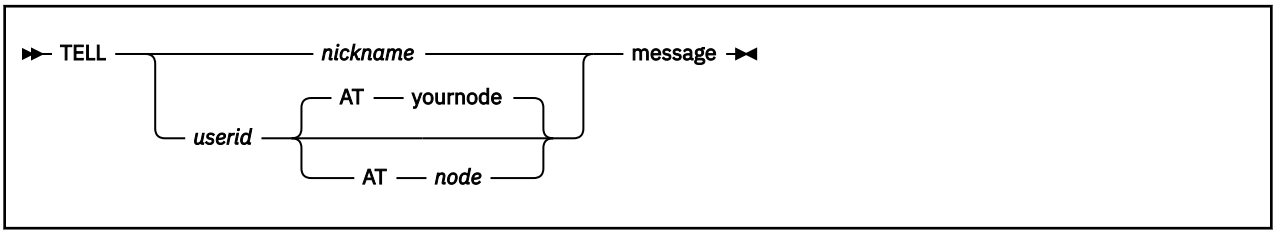
Messages and Return Codes

- DMS003E Invalid option: *option* [RC=24]
- DMS027E Invalid device *devtype* [for SYSaaa] [RC=24]
- DMS058E End-of-file or end-of-tape [RC=40]
- DMS105S Error *nn* writing file *fn ft fm* on disk or directory [RC=100]
- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS110S Error reading tap*n*(*vdev*) [RC=100]
- DMS335W Tap*n* [(*vdev*)] has been manually rewound and unloaded. Requested tape function may not have been executed. [RC=4]
- DMS420E NSL exit filename missing or invalid [RC=24]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in the Shared File System	“File Pool Server Messages” on page 1414
Errors in using a file	“File Error Messages” on page 1418

TELL



Authorization

General User

Purpose

Use the TELL command to send a message to one or more computer users on your computer or on other computers. 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 more information on creating a *userid* NAMES file, see "NAMES" on page 535.

Operands

nickname
userid
specifies one or more recipients to whom the message is to be sent. If the same recipient is specified more than once, the user receives only one copy of the message.

- A *nickname* is a short form of a user ID you have defined in your *userid* NAMES file, where *userid* is your user ID. This *nickname* may represent a single person (on your computer or another computer), or a list of several people. If a *nickname* cannot be found in the *userid* NAMES file, it is assumed to be a fully-specified user ID of someone on your system. For more information on nicknames, see "NAMES" on page 535.
- A *userid* is specified in the form *userid* AT *node*, which identifies a user (*userid*) on your system (yournode) or another system (*node*).

A user ID cannot be *AT* or *CC*.

You may freely intermix the *userid* and *nickname* forms to specify recipients.

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 *userid* AT *node* form.
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), the user 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. For more information, see "DEFAULTS" on page 153. For more information, on the MSGNOH, SMSG, and WARNING commands, see *z/VM: CP Commands and Utilities Reference*.
4. The TELL command uses the CP SMSG command to send messages, through RSCS, to users logged on to other computers (nodes). A warning message can 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
```

6. In an SSI cluster, if you issue the TELL command without specifying the node, and the recipient is a single-configuration virtual machine, the message will be sent to the cluster member where the recipient is logged on. If you omit the node and the recipient is a multiconfiguration virtual machine, the message will be sent only to the logon instance on the member where the TELL command was issued. If you specify the node for either type of user, the message will be sent only if RSCS is running on both members.

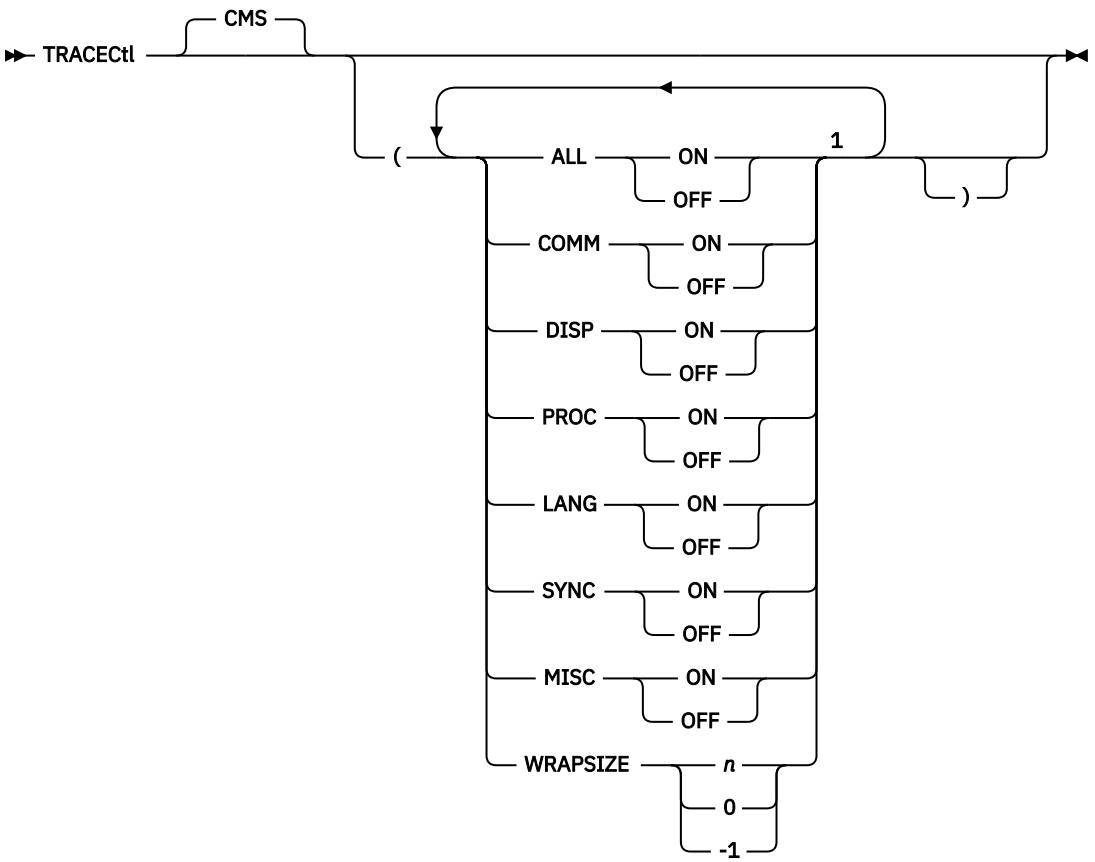
Messages and Return Codes

- DMS149E Userid *userid* not valid; no message has been sent [RC=32]
- DMS399E Tag too long for *nickname* in *userid* NAMES file [RC=88]
- DMS499E User not authorized to issue the *command* command [RC=40]
- DMS647E Userid not specified for *nickname* in *userid* NAMES file [RC=32]
- DMS648E Userid *name* not {found|resolved}; no message has been sent [RC=32]
- DMS653E Error executing *command*, rc=*rc* [RC=40]
- DMS676E Invalid character * for Network ID [RC=20]
- DMS1012E Node ID *node* not valid; no message has been sent [RC=32]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

TRACECTL



Notes:

[1] If an option is specified more than once, the last one is used.

Authorization

General User

Purpose

Use the TRACECTL command to initiate tracing in CMS for specific events. When tracing is set to ON, this starts the internal CMS tracing for the trace event(s) set, and controls tracing for the entire session. For more information on trace events, see *z/VM: CMS Application Multitasking*.

This command is useful for problem diagnosis in the CMS multitasking environment.

Options

One or more of the following options may be specified:

ALL
enables or disables tracing for all CMS trace events.

ON
sets tracing ON for all events.

OFF
sets tracing OFF for all events.

COMM
enables or disables tracing for Communication trace events.

ON
sets tracing ON for Communication events.

OFF
sets tracing OFF for Communication events.

DISP
enables or disables tracing for Dispatch trace events.

ON
sets tracing ON for Dispatch events.

OFF
sets tracing OFF for Dispatch events.

PROC
enables or disables tracing for Process Management trace events.

ON
sets tracing ON for Process Management events.

OFF
sets tracing OFF for Process Management events.

LANG
enables or disables tracing for Language-specific trace events.

ON
sets tracing ON for Language events.

OFF
sets tracing OFF for Language events.

SYNC
enables or disables tracing for Synchronization trace events.

ON
sets tracing ON for Synchronization events.

OFF
sets tracing OFF for Synchronization events.

MISC
enables or disables tracing for miscellaneous trace events.

ON
sets tracing ON for miscellaneous events.

OFF
sets tracing OFF for miscellaneous events.

WRAPSIZE
specifies how many trace events to retain if no eligible trace event monitor exists at the time the event is signaled. The valid values are -1 through 99999999.

50
a positive integer value of 50. This is the default if wrapsize is not specified. When the wrapsize value of 50 is exceeded, the oldest trace event is discarded to make room for the newest arrival.

n
a positive integer value greater than 0. When the wrapsize is exceeded, the oldest trace event is discarded to make room for the newest arrival.

0
No trace events are to be retained.

-1
Trace events continue to be retained until virtual storage is exhausted.

Usage Notes

1. During initialization, the following trace attributes are in effect:

   ```
   ALL OFF
   WRAPSIZE 50
   ```

   These values may be overridden when a user invokes the TRACECTL command with its own settings.
2. The TRACECTL options are processed in the order specified. For example, if the first option is ALL and is set to OFF, all trace categories are set off before processing subsequent options. As a result, in general, an option may be nullified by a subsequent option. Also, if an option is repeated, the first setting(s) of the option specified will be nullified by any subsequent setting of the option.
3. If you invoke an application that uses the TraceControl application programming interface, the trace event settings currently in effect can be changed.
4. For more information on how to obtain the current WRAPSIZE value and trace event(s) in effect, see "QUERY TRACECTL" on page 768.
5. Trace events enabled by the TRACECTL command can affect the trace data being collected by Event Services to produce accounting information. Accounting is controlled by the AccountControl application programming interface.
6. TRACECTL always results in an event definition of session scope and broadcast signals.
7. For more information on each trace event category being enabled, see the trace record formats in *z/VM: CMS Application Multitasking*.

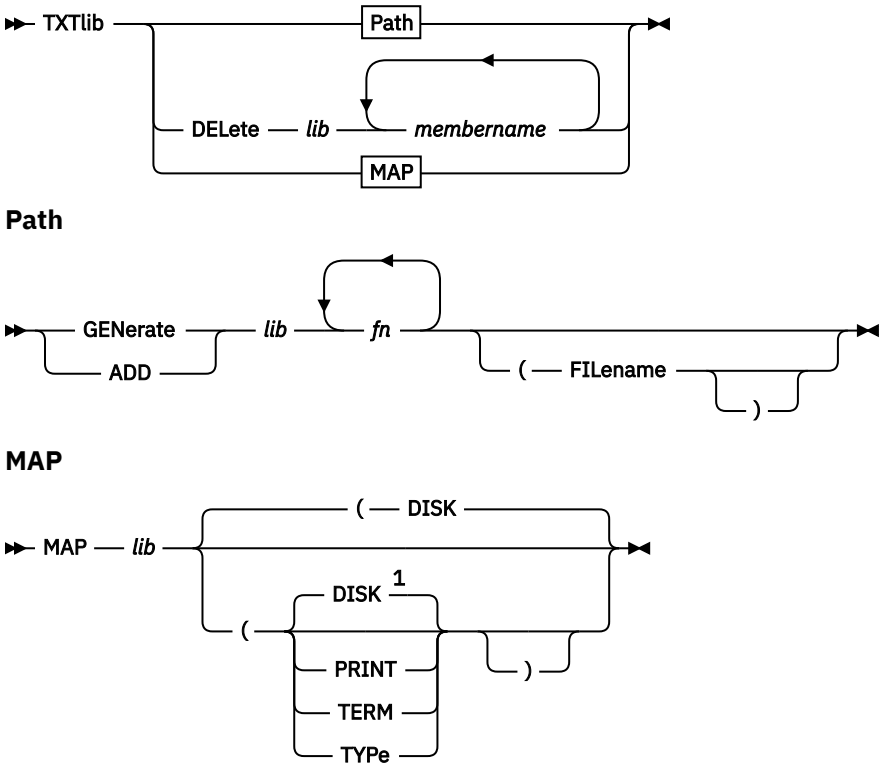
Messages and Return Codes

- DMS2011E WRAPSIZE value must be -1 or greater. [RC=8]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	"Command Syntax Error Messages" on page 1411

TXTLIB



Notes:

[1] PRINT and TERM erase the old MAP file, if one exists.

Authorization

General User

Purpose

Use the TXTLIB command to update CMS text libraries.

Operands

GENerate
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.

DELete
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.

If a TXTLIB DELETE command abends, temporary file 'TXTLIB CMSUT1 A1' can remain on the disk. You must delete or rename this file before executing another TXTLIB DELETE command. If the library

TXTLIB file that was being updated before the abend still exists, delete the 'TXTLIB CMSUT1' file; otherwise, rename it to the name of the library TXTLIB file that has been deleted.

MAP
lists the names (entry points) of TXTLIB members, their locations in the library, and the number of entries.

lib
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. Neither the library nor the file can be empty.

fn
specifies the name(s) of file(s) with file type(s) of TEXT you want to add to a TXTLIB.

membername
specifies the name(s) of TXTLIB member(s) you want to delete.

Options

Option A

FILename
indicates 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 "lib MAP A1", that contains a list of TXTLIB members on your minidisk or directory accessed as A. If a file of that name already exists, the old file is erased. This is the default.

PRINT
spools a copy of the TXTLIB map to the virtual printer.

TYPe
displays information about the TXTLIB on your terminal. (This option has the same function as the TERM option.)

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.
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 (TXTLIB DELETE) must be made on the member name. Other CMS commands will treat entry point name(s) the same as member name(s), which means they will search all member names and entry point names in sequential order.
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 MVS 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 can return erroneous entry points during dynamic loading.
9. The total number of entries in the TXTLIB file cannot exceed 6000. When this number is reached, an error message is displayed. The text library created includes all the text files entered up to (but not including) the one that caused the overflow.

   The total number of entry points in a member cannot exceed 4048. When this number is reached, an error message is displayed and the next text file (if there is one) is processed. The total number of ALIAS linkage editor control statements in a member cannot exceed 64. When this number is reached, an error message is displayed and the next text file (if there is one) is processed.
10. The PRINT and TERM options erase the old MAP file, if one exists.
11. A return code of 4 indicates 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 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 can produce unpredictable results.
12. If TXTLIB encounters a not valid text library, TXTLIB (except TXTLIB GENERATE) will stop processing and issue an error message.

    If TXTLIB encounters a not valid input TEXT file (that is, a variable record format or empty file, or a file with lrecl not equal to 80), it will stop processing, but any processing that occurred before the termination will still be valid.

Responses

When the TXTLIB MAP command is issued with the TERM or TYPE 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

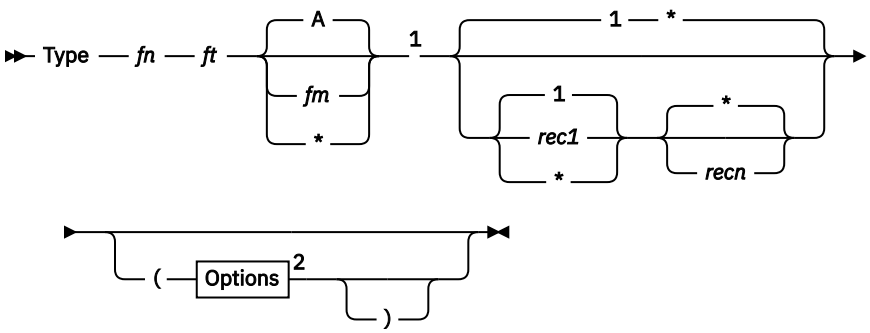
- DMS001E No {file|CSECT} name specified [RC=24]
- DMS002W File *fn ft [fm]* not found [RC=4 or 28]
- DMS003E Invalid option: *option* [RC=24]

- DMS013E Member *membername* not found in file *fn ft fm* [RC=32]
- DMS024E File *fn* [*ft fm*] already exists [RC=28]
- DMS037E Filemode *mode*[(*vdev*)] is accessed as read/only [RC=36]
- DMS056E File *fn ft [fm]* contains invalid [name|alias|entry|ESD] record formats [RC=32]
- DMS056W File *fn ft* contains invalid [name|alias|entry|ESD] record formats [RC=4]
- DMS069E Filemode *mode* not accessed [RC=36]
- DMS104S Error *nn* reading file *fn ft fm* from disk or directory [RC=31|55|70|76|99|100]
- DMS105S Error *nn* writing file *fn ft fm* on disk or directory [RC=31|55|70|76|99|100]
- DMS106S Number of entry names exceeds maximum of 6000; file *fn* TEXT not added [RC=88]
- DMS213W Library *fn* TXTLIB not created [RC=4]
- DMS213W Library *fn* TXTLIB not created, or erased if empty [RC=4]
- DMS213W Library *fn* TXTLIB has no members [RC=4]
- DMS257T Internal system error at address *addr* (offset *offset*)
- DMS1144E Implicit rollback occurred for work unit *workunitid* [RC=31]
- DMS1252T Rollback unsuccessful for file pool *filepoolid*

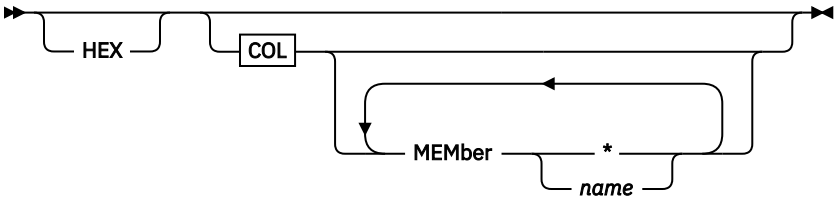
Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411
Errors in the Shared File System	“File Pool Server Messages” on page 1414
Errors in using a file	“File Error Messages” on page 1418
Errors in copying a file	“Messages and Return Codes” on page 90
Errors in erasing a file	“Messages and Return Codes” on page 217
Errors in renaming a file	“Messages and Return Codes” on page 826

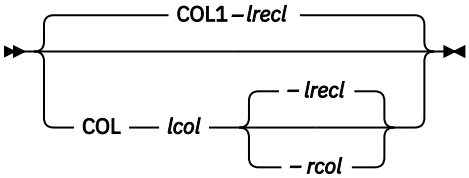
TYPE



Options



COL



Notes:

[1] The *fm* parameter is required if * is specified for *rec1*.
[2] You can enter Options in any order between the parentheses.

Authorization

General User

Purpose

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.

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.

rec1
is the record number of the first record to be displayed. This field cannot contain special characters. If rec1 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. 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

HEX
displays the file in hexadecimal format.

COL *lcol–rcol*
displays only certain columns of each record. Column *lcol* specifies the start column and *rcol* the end column of the field within the record that is to be displayed. The string *lcol–rcol* 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. If the file type is LISTING, the first position of each record sent to the terminal is changed to X'17', because the first position is assumed to contain a carriage control character.

MEMber *
MEMber *name*
displays member(s) of a library. If the format of the file is MACLIB or TXTLIB or LOADLIB, 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 can 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 line size (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
```

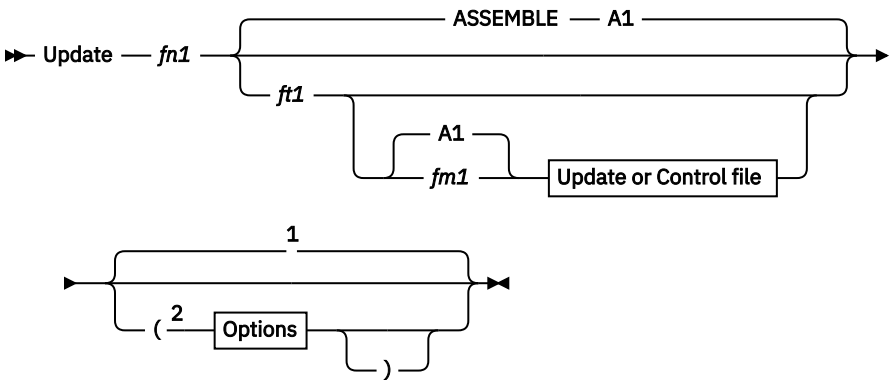
Messages and Return Codes

- DMS002E File [*fn* [*ft* [*fm*]]] not found [RC=28]
- DMS003E Invalid option: *option* [RC=24]
- DMS005E No *option* specified [RC=24]
- DMS009E Column *col* exceeds record length [RC=24]
- DMS013E Member *membername* not found in library [RC=32]
- DMS029E Invalid parameter *parameter* in the option *option* field [RC=24]
- DMS033E File *fn ft fm* is not a library [RC=32]
- DMS039E No entries in library *fn ft fm* [RC=32]
- DMS049E Invalid line number *nn* [RC=24]
- DMS054E Incomplete fileid specified [RC=24]
- DMS062E Invalid * in fileid [RC=20]
- DMS069E Filemode *mode* not accessed [RC=36]
- DMS104S Error *nn* reading file *fn ft fm* from disk or directory [RC=100]
- DMS1229I *fileid* is empty [RC=0]

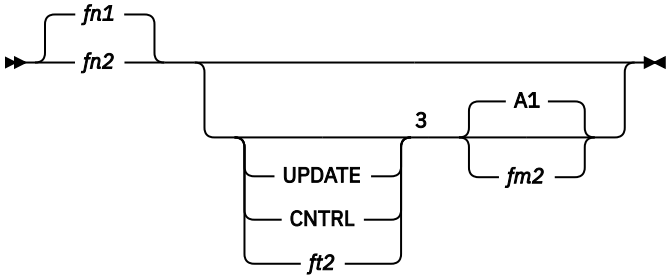
Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411
Errors in the Shared File System	“File Pool Server Messages” on page 1414
Errors in using a file	“File Error Messages” on page 1418

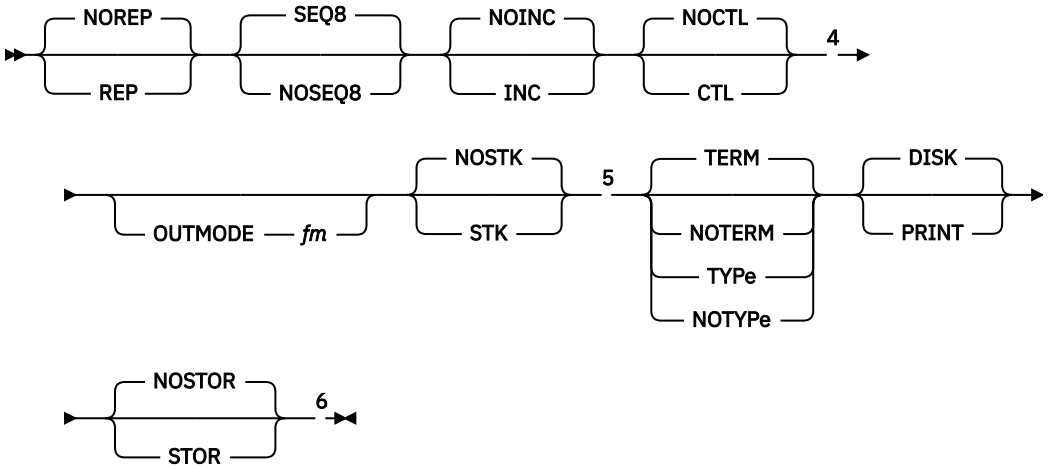
UPDATE



Update or Control file



Options



Notes:

[1] The defaults you receive appear above the line in the Options fragment.
[2] You can enter Options in any order between the parentheses.
[3] Update or control file is *fn1* UPDATE A1 if NOCTL is specified, *fn1* CNTRL A1 if CTL is specified.
[4] CTL implies INC.
[5] STK is only valid when CTL is specified and when UPDATE is entered from EXEC.
[6] STOR is the default when CTL is specified.

Authorization

General User

Purpose

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.

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. It cannot be an empty file.

fn2 ft2 fm2
is the file identifier of the update file or control file. If you specify the NOCTL option, this file must be an update file that lists the updates you choose to apply. The update file can 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. A control file cannot be an empty file.

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*). This is the default.

SEQ8
specifies the entire sequence field (the last eight columns of the file) contains an eight-digit sequence number on every record of source input. This is the default.

NOSEQ8
specifies 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. This is the default.

CTL
specifies *fn2 ft2 fm2* describes an update control file for applying multiple update files to the source input file.

NOCTL
specifies a single update file is to be applied to the source input file. This is the default.

OUTMODE *fm*
specifies 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, the files are put onto the disk or directory as outlined in in 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. This is the default.

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.) This is the default.

NOTERM
suppresses the display of warning messages at the terminal. However, error messages that stop the entire update procedure are displayed at the terminal.

TYPe
has the same function as the TERM option.

NOTYPe
has the same function as the NOTERM option.

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. This is the default.

PRINT
prints the update log file directly on the virtual printer. For more information, see Usage Note "1" on page 1094.

STOR
specifies the source input file is to be read into storage and the updates performed in storage before placing the updated source file on a disk or directory. This option is meaningful only for use with the CTL option, because the benefit of increased processing speed is realized when processing multiple updates. STOR is the default when you specify CTL.

NOSTOR
specifies no updating is to take place in storage. NOSTOR is the default when you apply single updates (for example, you do not specify CTL on the command line). This is the default.

Usage Notes

1. If the PRINT option is specified and the LPP option on the CP SPOOL command is 0 (LPP OFF), the number of lines per page for the update log file will be 55. If the LPP value is greater than 0, the number of lines per page for the update log file will be equal to the LPP value. For more information, see *z/VM: CP Commands and Utilities Reference*.

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.

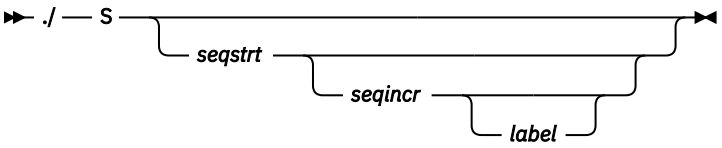
Control Statement Formats

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 renumbers 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:



Where:

seqstrt
is a 1-8 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 1-8 digit numeric field specifying the decimal increment for resequencing the output file. The default is the "seqstrt" 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.

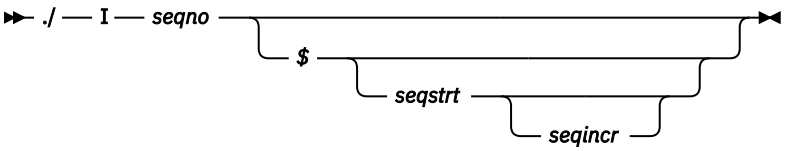
When the sequence control statement is the first statement processed, the sequence numbers in the source file are checked and warning message DMSUPD210W is issued for any errors. If the sequence control statement is processed after updates have been applied, no warning messages will be issued.

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 you may want to add to a source file.

The format of the INSERT control statement is:



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.

seqstrt
is a 1-8 digit numeric field specifying the first decimal increment for sequencing the inserted records.

seqincr
is a 1-8 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 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 $
```

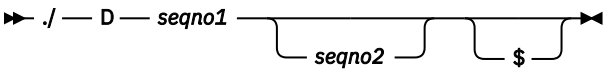
the defaulted *seqincr* is the maximum, 100, and the starting sequence number is 240100. UPDATE assumes SEQ8, so it sequences the inserted records, 00240100, 00240200, and so forth.

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:



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.

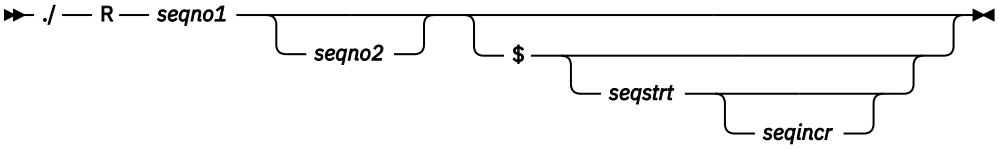
$
is an optional delimiter indicating the end of the control fields.

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*, 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:



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 the substituted records are to be sequenced incrementally.

seqstrt
is a 1-8 digit numeric field specifying the first decimal number to be used for sequencing the substituted records.

seqincr
is a 1-8 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'
```

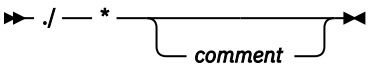
replace the existing statements numbered 38000 through 38500 with the new lines of code. As with the INSERT statement, UPDATE does not automatically resequence new lines. In addition, the dollar sign ($), *seqstrt*, and *seqincr* processing is identical with that for the INSERT statement.

COMMENT Statement

Lets you place comments in the update log file.

Note: 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:



Where:

***
specifies 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

This 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

These steps determine 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, 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, this message is displayed:

   ```
   DMS037E  Filemode mode [(vdev)] is accessed as read/only
   ```

2. If the disk or directory on which the original source file resides is read/write, 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, UPDATE places the output files on that particular read/write disk or directory.
4. If neither of the last two steps is successful, UPDATE places the output files on the disk or directory accessed as A.

Applying a Single Update

Say 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 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 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 you want to apply to a source file. The control file itself does not contain the actual UPDATE control statements.

Say you have two update files, *fn* UPDTABC and *fn* UDPTXYZ, and 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. It may not 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 UPDTXYZ
```

- *TEXT*, *TWO*, and *ONE* are update level identifiers. An update level identifier is the first word in a line and can be from 1-5 characters long. z/VM updating procedures such as the VMFASM EXEC use these identifiers to locate and identify text decks produced by multi-level updates.
- *MACS* must be the first uncommented record in the control file. It contains an update level identifier (TEXT) and, optionally, macro library (MACLIB) file names, subject to the logical record length. MACLIB names must be separated by blanks. 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. The total number of MACLIBS permitted in a control file is 63.

  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.

You may specify comments on MACS records by means of an asterisk. Any information beyond the asterisk will be treated as a comment and will not be passed on when you specify the STK option.

- *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 these:

```
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
```

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 can 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.

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 these:

```
TEXT MACS CMSLIB
TWO UPDTABC
ONE UPDTXYZ
TEXT AUXLIST
```

- *AUX* in the file type AUXLIST indicates 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.

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.

When you issue:

```
update fn (ctl
```

UPDATE looks in the control file at the bottom entry, TEXT AUXLIST. Because 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: 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 yyyyy AUX
```

in the control file.

Where:

xxxxx
specifies the update level identifier and

yyyyy
specifies the last five characters following AUX in the file type.

For example, the record:

```
TEXT LIST AUX
```

identifies the auxiliary file *fn* AUXLIST.

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 can 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.

This is an example of a control file showing 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 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 the additional file types do exist, UPDATE ignores that entire entry and proceeds to the next entry in the control file. The files that indicate additional file types are preferred because UPDATE does not use the file the first file type indicates. For example, assume you want to update the file SAMPLE ASSEMBLE using the updates in SAMPLE AUXTEST. To update SAMPLE ASSEMBLE, use this control file (MYMODS 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. Because 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 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 not a valid 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 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 DMSGPI DMSOM  * basic system MACLIBS
LCL  AUXLCL * Local Mods
TEXT AUXVM  * PTF service
```

and the DMSABC AUXVM and DMSABC AUXLCL auxiliary control files both contain updates, and the UPDATE command appears in a REXX exec as follows:

```
/*************************************************/
/* Update module DMSABC ASSEMBLE using control file  */
/* DMSVM CNTRL and stack the update level identifier */
/* and the list of MACLIBs from the MACS records.    */
/* Note: No error checking is performed here.        */
/************************************************* */
'UPDATE DMSABC ASSEMBLE A DMSVM CNTRL A (CTL STK'
Parse Upper Pull star updtlvl
Parse Upper Pull star maclibs
```

the following variables will be set:

Variable
Value

star
*

updtlvl
LCL

maclibs
DMSGPI DMSOM

The library list may be useful to establish macro libraries in a subsequent GLOBAL command within the exec. If no update files are found, UPDATE stacks the update level identifier on the MACS record.

Missing Update Files

The following return codes are displayed when there are missing update files.

- RC=0. This is returned when the only missing update files are files that are referenced directly through control files, and there was at least one update file found.
- RC=12. This is returned if any of the update files referenced from an AUX file are not found, regardless of whether any other update files are found.
- RC=40. This is returned if no update files are found and either AUX files are not used or there are no update files referenced by them. An RC=40 is also returned if AUX files are used, but there are no update files referenced.

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.

DMS177I 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:

4
Sequence errors were detected in the original source file being updated.

8
Sequence errors, which did not previously exist in the source file being updated were introduced in the output file during the updating process.

12
Any other warning error detected during the updating process. Such errors include not valid 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.

```
DMS178I  Updating fn ft fm
            Applying fn ft fm
            Applying fn ft fm
            Applying fn ft fm (empty file)
                .
                .
                .
```

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. If the update file is empty, the text with "(empty file)" is displayed. The updating process continues.

DMS304I 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.

DMS1229I *fn ft fm* is empty
This message is displayed when the auxiliary control file is an empty file.

DMS180W Missing PTF file *fn ft fm* RC=12
In the event 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

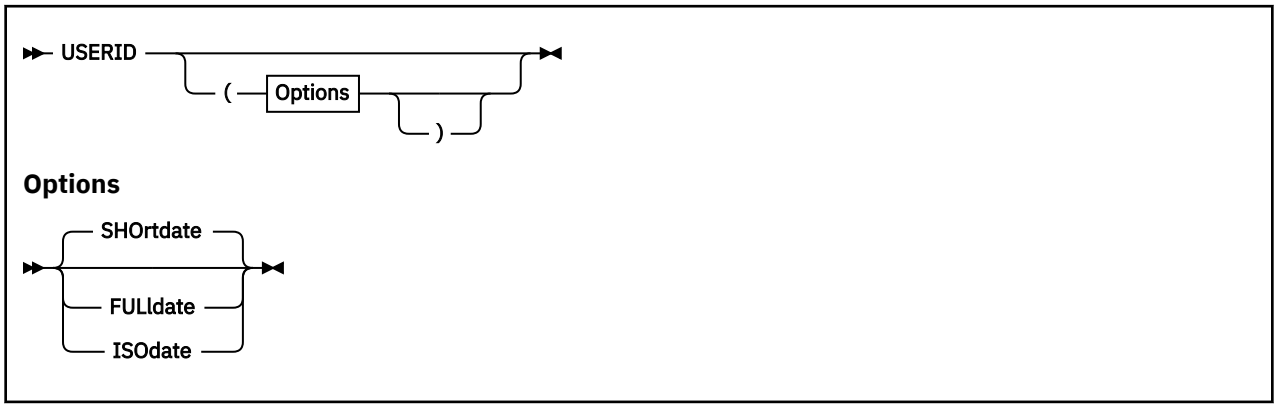
- DMS001E No file name specified [RC=24]
- DMS002E File [*fn* [*ft* [*fm*]]] not found [RC=28]
- DMS003E Invalid option: *option* [RC=24]
- DMS007E File *fn ft fm* is not fixed, 80-character records [RC=32]
- DMS007E File *fn* does not have the same format and record length as *fn* [RC=32]
- DMS007E File *fn ft fm* is not fixed record format [RC=32]
- DMS007E File *fn ft fm* does not have a logical record length greater than or equal to 80 and less than or equal to 255 [RC=32]
- DMS010W Premature EOF on file *fn ft fm*--sequence number *seqno* not found [RC=12]
- DMS024E File *fn ft fm* already exists [RC=28]
- DMS048E Invalid filemode *mode* [RC=24]
- DMS065E *option* option specified twice [RC=24]
- DMS066E *option1* and *option2* are conflicting options [RC=24]

- DMS069E Filemode *mode* not accessed [RC=36]
- DMS070E Invalid parameter *parameter* [RC=24]
- DMS104S Error *nn* reading file *fn ft fm* from disk or directory [RC=100]
- DMS105S Error *nn* writing file *fn ft fm* on disk or directory [RC=100]
- DMS174W Sequence error introduced in output file: *seqno1* to *seqno2 seqno1* to *seqno2* [RC=8]
- DMS176W Sequencing overflow following sequence number *seqno* [RC=8]
- DMS179E Missing or invalid MACS card in control file *fn ft fm* [RC=32]
- DMS181E No update files were found [RC=40]
- DMS182W Sequence increment is zero [RC=8]
- DMS183E Invalid {CONTROL|AUX} file control card [RC=32]
- DMS184W ./S not first card in update file--ignored [RC=12]
- DMS185W Invalid character in sequence field *seqno* [RC=12]
- DMS186W Sequence number *seqno* not found [RC=12]
- DMS187E Option STK invalid without CTL [RC=24]
- DMS207W Invalid update file control card [RC=12]
- DMS210W Input file sequence error: *seqno1* to *seqno2* [RC=4]
- DMS299E Insufficient storage to complete update [RC=41]
- DMS300E Insufficient storage to begin update [RC=41]
- DMS361E Disk *mode* is not a CMS disk [RC=36]
- DMS1213W Update *fn ft fm* is an UPDATE SHELL [RC=12]
- DMS1229E *fn ft fm* is empty [RC=32]
- DMS1259E File pool *filepoolid* has run out of physical space in the storage group [RC=100]
- DMS1262E Error *nn* opening file *fn ft fm* [RC=*nn*]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411
Errors in using a file	“File Error Messages” on page 1418
Errors in using the Shared File System	“File Pool Server Messages” on page 1414

USERID



Authorization

General User

Purpose

Use the USERID command to return the user ID and other information via the CMS console stack.

Options

SHOrtdate
displays the dates in *mm/dd/yy* format, where *mm* is the month, *dd* is the day of the month, and *yy* is the 2-digit year. This is the default.

FULldate
displays the dates in *mm/dd/yyyy* format, where *mm* is the month, *dd* is the day of the month, and *yyyy* is the 4-digit year.

ISOdate
displays the dates in *yyyy-mm-dd* format, where *yyyy* is the 4-digit year, *mm* is the month, and *dd* is the day of the month.

Usage Notes

1. The format of the stacked line is:

Token	Content	Description
1	*USERID	(viewed as a comment by cms
2	userid	whatever the userid is
3	reserved	%
4	date	mm/dd/yy for SHORTDATE, mm/dd/yyyy for FULLDATE, yyyy-mm-dd for ISODATE
5	time	hh:mm:ss
6	day	three characters (MON, TUES, etc.)
7	day-of-week	(numeric-Monday=1, Tuesday=2, and so on)

2. In addition to the USERID command, you can also use the IDENTIFY command to obtain similar results.

Messages and Return Codes

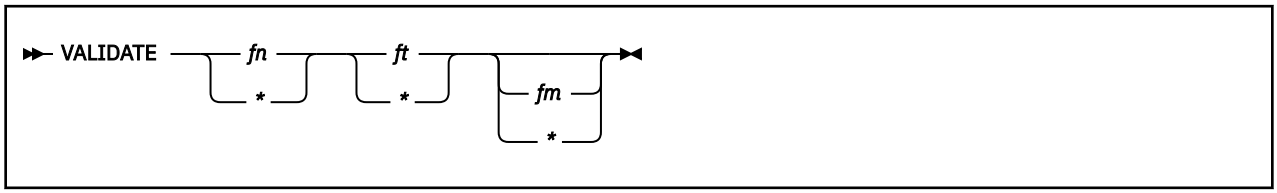
Return codes:

RC
Meaning

0
Operation complete — no errors

100
Explanation complete (when ‘?‘ specified)

VALIDATE



Authorization

General User

Purpose

Use the VALIDATE command to verify the syntax of a file identifier (file name, file type, file mode). If you specify the file mode, the VALIDATE command specifies the disk or directory is accessed.

Note: The VALIDATE command does NOT check that the file name and file type are greater than eight characters. This command only uses the first eight characters of the file name and file type entered.

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 1-8 characters. The valid characters are A-Z, a-z, 0-9, $, #, @, +, - (hyphen), : (colon), and _ (underscore).

   Note: The file name and file type are truncated after eight characters.
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, messages DMS069E and DMS070E 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.

   Note: The type of disk accessed is not checked for example, CMS, DOS, or OS.

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.

Responses

The CMS ready message indicates the specified file identifier is valid and the file mode is an accessed disk or directory or was specified as *.

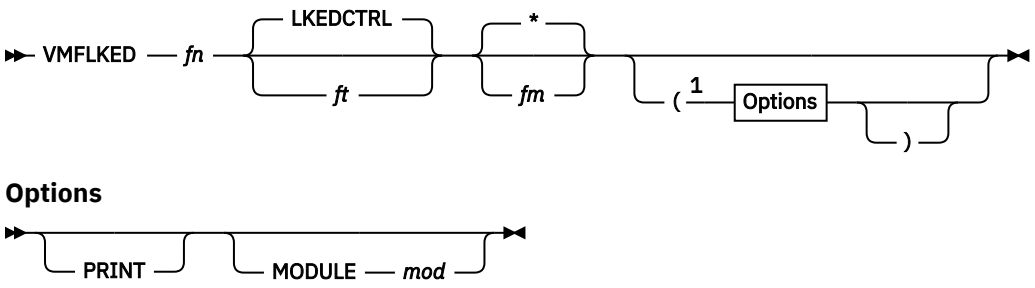
Messages and Return Codes

- DMS048E Invalid filemode *mode* [RC=24]
- DMS054E Incomplete fileid specified [RC=24]
- DMS062E Invalid character *char* in fileid *fn ft* [RC=20]
- DMS069E Filemode *mode* not accessed [RC=36]
- DMS070E Invalid parameter *parameter* [RC=24]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

VMFLKED



Notes:

[1] You can enter Options in any order between the parentheses.

Authorization

Service Installer

Purpose

Use the VMFLKED command to invoke the CMS LKED command to link-edit modules into a LOADLIB. VMFLKED uses the general CMS search order when searching for TEXT files.

Operands

fn
specifies the file name of the input control file. You must specify a file name.

LKEDCTRL
specifies the default file type of the input control file.

ft
specifies the file type of the input control file.

****
specifies the default file mode of the input control file.

fm
specifies the file mode of the input control file.

Options

PRINT
prints out a hardcopy of the linkage-editor output.

MODULE *mod*
specifies only those members of the LOADLIB that include *mod* are to be link-edited.

VMFLKED Processing

VMFLKED reads the specified LKEDCTRL file and expects to find option records (if any are needed) followed by linkage-editor input records. Multiple groups of options followed by linkage-editor input can be combined in a single LKEDCTRL file, see Figure 51 on page 1111.

1. VMFLKED processes the option records, which are identified by '%' in column one.
2. VMFLKED processes linkage-editor input records when it finds a nonoption and noncomment record.

   When VMFLKED finds:

- An INCLUDE record, the process adds the record to the linkage-editor input file and issues a FILEDEF for the TEXT file.

  If the text deck contains history records and the LET option of the LKED command was **not** specified in the previous %LEPARMS option record in the control file, VMFLKED strips the history records out of the text deck before the text deck is link-edited. VMFLKED copies the text deck minus the history records to file mode A as a temporary file ($ concatenated to the file type and a file mode number of 3). The history records are placed into the LKEDIT file that contains the linkage-editor map.
- A NAME record, the process adds the record to the linkage-editor input file and invokes the linkage editor.
- A comment record before the ESD card, VMFLKED strips the comment from the text. Comments begin with an asterisk (*). Comments removed from the TEXT file are displayed in the linkage-editor map, which provides a record of the fixes installed when linkedit was performed.
- Any other record, the process adds the record to the linkage-editor input file.

3. VMFLKED continues with Step 1 to process the next group of option records (if any).

Responses

Input Files

***fn* TEXT**
The name of the TEXT file. When VMFLKED finds an INCLUDE statement in the linkage-editor input control file, it issues a FILEDEF for that TEXT file. The linkage editor reads this TEXT file.

***fn* LKEDCTRL**
A modified linkage-editor file. INCLUDE cards include TEXT files, and you cannot use them to include files from a library. Any record containing an asterisk (*) in column one is commentary and VMFLKED ignores it.

VMFLKED also recognizes special control (option) records in the linkage-editor control file. These records, which must begin with a percent symbol (%) in column one, may only be located between linkage-editor input files (that is, groups of them can be at the beginning of the file or following a NAME record). Table 50 on page 1109 describes the special control records in the linkage-editor control file.

Table 50. Linkage-Editor Control File Special Control (Option) Records

Record	Parameter and Function
%CONTROL	**control_file_name** This record indicates the file types for linkage-editor input files should be taken from a control file. The name of the control file is specified on the %CONTROL record and the file type is CNTRL. When the %CONTROL record is read, VMFLKED reads the control file and uses the first "word" of each line of the file to build an array of file types. The first line of the file indicates the default file type. Each of the file types is checked and if it is not TEXT, the characters TXT are put at the beginning. When VMFLKED finds an INCLUDE card, it searches the array. VMFLKED uses the first file type from the array that matches an existing file. This option is in effect until a %NOCONTROL record is found.
%NOCONTROL	This record indicates the file type for linkage-editor input files should not be taken from a control file.
%LIBRARY	**load_library_name** This record changes the LOADLIB to be used and the name on the LKEDIT listing. The default for the LOADLIB name is the file name of the LKEDCTRL file.

Table 50. Linkage-Editor Control File Special Control (Option) Records (continued)

Record	Parameter and Function
%LEPARMS	**link-edit parameters** This record sets the link-edit parameters used in each link-edit step. If no parameters are specified, none are used.
%MAXRC	**maximum_valid_return_code** This record sets the maximum valid return code. VMFLKED checks this value when it ends. If the highest return code is higher than this value and is not listed on the %ACCEPTRC record, the exec issues a warning message.
%ACCEPTRC	**return codes** This record lists the acceptable return codes for VMFLKED processing. (The return codes are usually higher than entry on the %MAXRC entry). VMFLKED checks the values after each link-edit. If the return codes after any link-edit is higher than the %MAXRC record and not specified on the $ACCEPTRC record, VMFLKED issues a warning message at the end of its processing.
%IGNORE	This record causes the exec to bypass the warning message for missing TEXT files. Once specified, this record takes effect for all subsequent link-edits (or until a %NOIGNORE record is found).
%NOIGNORE	This record causes a warning message to be issued if there is a missing TEXT file. Once specified, this record takes effect for all subsequent link-edits (or until a %IGNORE record is found).
%ERASE	This record erases LOADLIB and LKEDIT files. This is useful when you rebuild the LOADLIB because it keeps the LOADLIB and LKEDIT files as small as possible. If this record is not specified, the LOADLIB and LKEDIT files are not erased. **Note:** The %ERASE control statement takes effect immediately, and erases the current LOADLIB. The LOADLIB is not erased if you use the MODULE option. (The LOADLIB can be changed using the %LIBRARY control statement).

Output Files

***fn* LOADLIB A**
The main output from the linkage editor. This file contains the link-edited load modules.

***fn* LKEDIT A**
The file that contains the linkage-editor map for all modules. It also contains any history records VMFLKED has stripped out of the text decks.

***fn* $*ft* A3**
A temporary copy of the text deck minus history records VMFLKED creates if the LET link-edit option is not specified in the previous %LEPARMS record in the LKEDCTRL file. This file could be left on file mode A if LKED fails during processing.

Figure 51 on page 1111 is an example of a LKEDCTRL file.

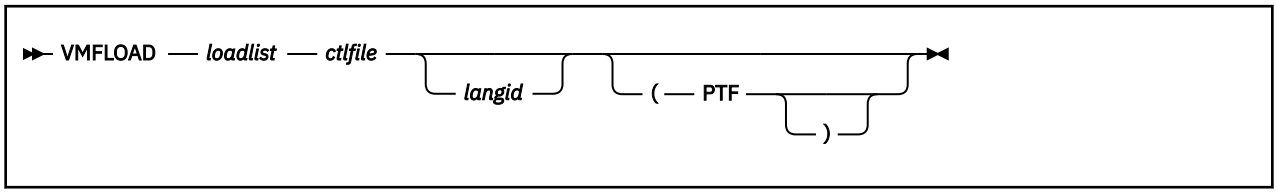
```
%CONTROL YOURCTRL
%LIBRARY NCCF
%ERASE
%MAXRC 4
%ACCEPTRC 12 14
%LEPARMS NCAL LIST XREF LET RENT
  INCLUDE  DSIZDST
  INCLUDE  DSIZSHP
  INCLUDE  DSILUTRM
  ORDER    DSIZDST
  ENTRY    DSIZDST
  NAME     DSIZDST(R)
%LEPARMS NCAL LIST XREF LET REUS
  INCLUDE  DSIVMCMD
  INCLUDE  DSIVMMSG
  ORDER    DSIVMCMD
  ENTRY    DSIVMCMD
  NAME     DSIVMCMD(R)
%IGNORE
%LEPARMS NCAL LIST XREF LET RENT
  INCLUDE  DSIPRTVM
  ENTRY    DSIPRTVM
  NAME     DSIPRTVM(R)
```

Figure 51. Example of a LKEDCTRL File

Messages and Return Codes

- DMS002E File *fn ft* [*fm*] not found [RC=28]
- DMS003E Invalid option: *option* [RC=8]
- DMS005E No {LKEDCTRL|MODULE} specified [RC=24]
- DMS010E Premature EOF on file *fn ft* [RC=12]
- DMS065E *parameter* parameter specified more than once [RC=100]
- DMS842E No {control|library} file name found in *fn ft* [*fm*]
- DMS843I An invalid control record was found and ignored:
- DMS844E No linkedit performed
- DMS845W Errors were encountered during the link edit processing that will probably make the loadlib unusable.
- DMS846I LKED *target_module* into *library*, rc=*rc*

VMFLOAD



Purpose

The VMFLOAD command generates a new CMS nucleus, z/CMS nucleus, or GCS nucleus. The VMFLOAD program uses a load list file, a control file, and an optional national language identifier to produce a punch file that contains several object modules.

Operands

loadlist
is the file name of the load list file. The file type must be EXEC. This file contains the names of the object modules to be punched. VMFLOAD punches the modules in the order specified, beginning at the top of the load list. VMFLOAD punches each module with a header card. For a list of the IBM-supplied load lists, see usage note "4" on page 1114.

A sample entry in a load list might look like this:

```
&CONTROL OFF
&1 &2 &3 fn [ft] [(LANG]
```

where:

fn
is the file name of the module to be punched.

ft
is the file type of the module to be punched.

LANG
is a special option used for national language-related files (message respositories, parser tables, and synonym tables). If you also specify a language ID on the VMFLOAD command, VMFLOAD does special processing with these entries to determine the actual file ID of the module to be punched.

ctlfile
is the file name of the control file. The file type must be CNTRL. This is usually the same control file used to apply updates to modules using the VMFASM or UPDATE command. The control file identifies the highest level object module available, if VMFLOAD cannot determine a specific file type from the load list entry. For a list of the IBM-supplied control files, see usage note "5" on page 1114.

langid
is the language identifier for national language-related files to be loaded. If you specify a language ID on the command line, VMFLOAD does special processing to determine the file ID of any module in the load list whose entry contains the LANG option. For a list of the supported national languages and their IDs, see usage note "6" on page 1115.

PTF
directs VMFLOAD to search for PTF-numbered text decks. You must use this option if you want to include PTF-numbered decks in your nucleus. If this option is not specified, VMFLOAD searches for text decks using the level identifier in the control file.

VMFLOAD Processing

VMFLOAD processes the entries in the load list from top to bottom, as follows:

1. If you specify the PTF option, VMFLOAD calls the VMFLDS MODULE to locate the text decks by PTF number. VMFLDS:
   a. Creates a temporary load list called $$$TLL$$ EXEC
   b. Processes each entry in the supplied load list as follows:
      - If the entry contains a file type, VMFLDS searches for a text deck with that file type.
      - If the entry does not contain a file type, VMFLDS must determine the file type:
        i) If you specified a language ID in the VMFLOAD command, VMFLDS checks for the LANG option in the load list entry. If the LANG option is present, and the file name in the entry is six characters, VMFLDS uses the language ID to obtain the corresponding country code from the VMFNLS LANGLIST file. If the country code exists, VMFLDS appends the country code to the end of the file name.
        ii) VMFLDS looks at the top entry in the control file for an AUX record, constructs a file type from the AUX file type, using the PTF number (TXT*ptfnum*) or the equivalent local entry (TXT*xxxxx*), and searches for a text deck with that file type.
        iii) If the control file entry does not contain an AUX file type or if VMFLDS cannot find the indicated text deck, VMFLDS constructs a file type from the level identifier (TXT*levelid*) and searches for a text deck with that file type.
        iv) If VMFLDS cannot find the text deck, it looks at the next control file entry.
        v) If VMFLDS exhausts all control file entries without finding the text deck, it uses the file type on the MACS record.
        vi) If VMFLDS still cannot find the text deck, it searches for the file type TEXT.
        vii) If VMFLDS still cannot find the text deck, and the entry has the LANG option, VMFLDS searches for an object file with a six-character file name (that is, without the country code) and a file type of TXT*langid*.
   c. Processes the text decks:
      - If VMFLDS cannot find a text deck, it displays a message at the console.
      - If VMFLDS locates a text deck, it:
        i) Writes the file name and file type of the text deck in the temporary load list.
        ii) Checks to see if the file is a full text deck or a text deck shell (containing only requisite information). If the file is a shell, VMFLDS issues a warning message.
        iii) Checks that the comments in the front of the text deck match the AUX file. If there is a mismatch, VMFLDS issues a warning message.
2. VMFLOAD issues the command:

   ```
   SPOOL PUNCH CONT
   ```

   to ensure that the punched files appear as one deck.
3. VMFLOAD processes each entry in the load list (the supplied load list or the load list created by VMFLDS in step “1” on page 1113) as follows:
   - If the load list entry contains a file name and file type, VMFLOAD searches for that file.
   - If the load list entry does not contain a file type, VMFLOAD must determine the file type:
     a. If you specified a language ID in the VMFLOAD command, VMFLOAD checks for the LANG option in the load list entry. If the LANG option is present, and the file name in the entry is six characters, VMFLOAD uses the language ID to obtain the corresponding country code from the VMFNLS LANGLIST file. If the country code exists, VMFLOAD appends the country code to the end of the file name.

b. VMFLOAD obtains the update level identifier from the control file and searches for the file type TXT*levelid*.

   Remember that updates applied to source files are applied from the bottom of the file towards the top. Therefore, VMFLOAD searches the control file from the top towards the bottom to locate the most recent update level.

   When determining the file types of object modules to punch, VMFLOAD ignores records that have an update level identifier of PTF and searches for the next lower level identifier.

   For example, if a control file contains the following records:

```
TEXT MACS DMSGPI DMSOM
LOCAL FIX1
SPEC  AUX1111
PTF   C12567DM
IBM1  AUXVM
```

   then, for each entry in the load list, the VMFLOAD search order is:

```
fn TXTLOCAL
fn TXTSPEC
fn TXTIBM1
```

c. If none of the above file types exist for the load list entry, VMFLOAD searches for file name TEXT.

d. If VMFLOAD cannot find a TEXT file, and the entry contains the LANG option, VMFLOAD searches for an object file with a six-character file name (that is, without the country code) and a file type of TXT*langid*.

e. If VMFLOAD still cannot find the object file, it displays a message and continues processing with the next entry in the load list.

4. When VMFLOAD locates an object module, VMFLOAD punches the module, then continues processing the next entry in the load list.
5. When all the object modules are punched, VMFLOAD issues the commands:

```
SPOOL PUNCH NOCONT
CLOSE PUNCH
```

Usage Notes

1. VMFLOAD requires a virtual machine with at least 512 KB.
2. VMFLOAD is called by the VMFBLD EXEC.
3. Before invoking VMFLOAD, you might want to enter one of the following commands:

```
spool pun to *
spool pun to userid
```

   to transfer the punched output as a reader file to your own virtual machine or to another virtual machine. If you want to do any additional controls, you should write an exec procedure to do the controls and invoke VMFLOAD from that exec.

4. z/VM supplies the following load lists for building a CMS, z/CMS, or GCS nucleus:

Load List	Usage
CMSLOAD EXEC	CMS nucleus
ZCMSLOAD EXEC	z/CMS nucleus
GCTLOAD EXEC	GCS nucleus

5. z/VM supplies the following control files for building a CMS, z/CMS, or GCS nucleus:

Control File	Usage
DMSVM CNTRL	CMS nucleus
DMSVMZ CNTRL	z/CMS nucleus
GCTVM CNTRL	GCS nucleus

6. The language IDs for national languages other than mixed-case American English are:

Country Code	Language ID	Language
B	UCENG	Uppercase English

7. After you have punched a new nucleus with VMFLOAD, you can use the MOVEFILE command to move the nucleus to tape, or, if the nucleus is in your virtual card reader, you can IPL it by entering:

```
ipl 00c
```

   When you IPL the virtual card reader, the loader is read first, and it loads the rest of the object modules. If the loader is successful, the nucleus is written on minidisk, and the load map is spooled to the virtual printer. If you want to preserve a copy of the load map, you should spool your printer to your virtual card reader before you read in the file.

8. To locate files, VMFLOAD searches all of your accessed minidisks and SFS directories using the standard search order, A through Z.

Input and Output Files

Type	File ID	Purpose
Input	*loadlist* EXEC	The load list file, which contains the file names and optionally the file types and/or LANG options for the object modules to be punched.
Input	DMKLD00E LOADER	The loader, which should be the first entry in the load list file.
Input	*ctlfile* CNTRL	The control file.
Input	*object* TEXT	An object module to be punched.
Input	*object* TXT*levelid*	An object module to be punched.
Output	*fn ft*	The load deck punched to your virtual machine. It has the same file ID as the last object module that was punched.

Responses

```
SYSTEM LOAD DECK COMPLETE
```

All the files in the load list have been punched.

```
INSUFFICIENT OR INVALID ARGUMENTS
```

The command line was incorrectly entered.

```
NO CONTROL FILE
```

The control file could not be located.

```
ERROR IN CONTROL FILE
```

The control file contains an invalid record.

```
NO LOAD LIST
```

The load list could not be located.

```
ERROR IN LOAD LIST
```

The load list contains an invalid record.

```
fn ft NOT FOUND
```

No text file was found.

```
ERROR ON PUNCH
```

An error occurred while punching a file.

Messages and Return Codes

z/VM: VMSES/E Introduction and Reference shows the format of VMSES/E messages and lists the identifiers used by each VMSES/E exec.

- VMFSLD1832W Text deck *fn ft* is included in the *name* build list currently being processed but the text deck cannot be found.
- VMFSLD1897E A problem occurred reading file *fn ft* while function *function* was executing.
- VMFSLD1898E A record cannot be written to file *fn ft fm*.

PI

Return codes issued by the VMFLOAD command may be returned to a user exit. See the description of the :USEREXIT tag in *z/VM: VMSES/E Introduction and Reference* for information about user exits.

RC
Meaning

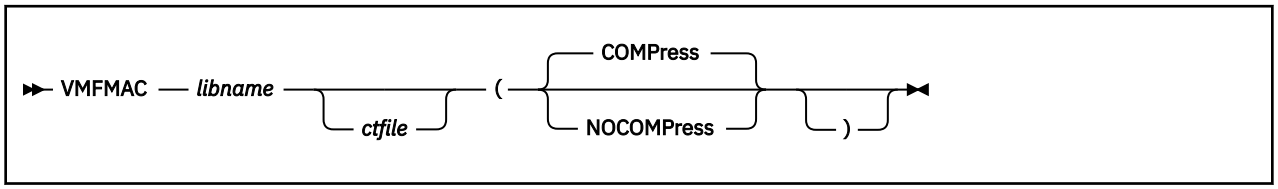
0
Command completed successfully.

4
Command completed with one or more warning conditions.

100
Command failed because of an external error.

PI end

VMFMAC



Authorization

Service Installer

Purpose

Use the VMFMAC command to update the macro libraries (MACLIBs). If you specify a control file, VMFMAC calls the CMS UPDATE command to update the specified copy of macro files listed in the control file and then builds a new macro library from the updated versions of those files.

Note: Copy files are collections of assembler statements commonly used by many modules. In source listings, the assembler includes copy files by a "COPY *copyfilename*" statement. They are similar to macro definitions, because they reside in libraries; but they contain no substitution data. z/VM uses copy files to define control block DSECTS, data areas, and tables.

Operands

libname
is the file name of the macro library to be updated and the EXEC file that lists the names of the library members. The entries in *libname* EXEC must be in the following format:

```
&1 &2 fn1
&1 &2 fn2
⋮
&1 &2 fnn
```

fn1, *fn2*, and *fnn* are the file names of the macro and copy files to be updated and included in the macro library. They must have a file type of MACRO or COPY.

ctlfile
is the file name of an optional control file that applies the updates. The file type must be CNTRL.

COMPress
specifies that comment lines in the source file that start in column 1 and begin with *.*!* should be removed from the macro source when the macro is included in a macro library.

Note: You identify these extraneous internal macro comments (for instance, a prolog), by putting an exclamation point (*!*) after the *.** that begins the comment.

NOCOMPress
specifies comments are not to be removed from the macro source when the macro is included in a macro library.

Usage Notes

1. When VMFMAC adds files with MACRO file types to a MACLIB, the EXEC takes the member name from the macro prototype statement. When VMFMAC adds files with COPY file types to a MACLIB, the EXEC follows these rules:

   a. If the member name is from the COPY file and updates were found, the member name is $*filename*.

b. If the member name is from the COPY file and no updates were found, the member name is *filename*.

c. If you include a *COPY statement as the first record in the file, the member name is that designated on the *COPY statement. The format of the *COPY statement is:

```
*COPY membername
```

2. If errors occur during VMFMAC processing, consult the NEWMAC COPY file printed by VMFMAC. If you can correct the errors involving one or two macro or copy files, you can use the MACLIB command to add these members to NEWMAC MACLIB. Then enter the following commands:

```
erase libname maclib
rename newmac maclib fm libname = =
```

The current *libname* MACLIB is erased, and NEWMAC MACLIB is renamed *libname* MACLIB.

3. If any accessed minidisk contains a MACRO file and a COPY file with the same file name, the MACRO version is used.

VMFMAC Processing

1. VMFMAC locates *libname* EXEC and the control file, if you specified one. It also erases any existing files named NEWMAC MACLIB and NEWMAC COPY. VMFMAC begins reading the macro or copy file names from the EXEC, beginning at the bottom.
2. If you specify a control file, for each entry in the *libname* EXEC, VMFMAC:

   a. Invokes the UPDATE command with the CTL and OUTMODE A1 options to apply the updates specified in the control file.

      The UPDATE command stacks information from the control file in the console stack and prints the update log file. The OUTMODE A1 option specifies the files created by the UPDATE command are written on file mode A with a file mode number of 1 (A1).

   b. Adds the updated macro or copy file ($*fn* MACRO or $*fn* COPY) to the macro library NEWMAC MACLIB.

   c. Adds the *fn* UPDATES file created by the UPDATE command to the file NEWMAC COPY.

   d. Erases $*fn* MACRO or $*fn* COPY and erases *fn* UPDATES.
3. If there are no update files corresponding to a macro or copy file specified in *libname* EXEC, the macro or copy file is added to NEWMAC MACLIB in its current form. If you specify a control file, the file NEWMAC COPY, which contains a history of the updates applied by VMFMAC, is added to NEWMAC MACLIB.
4. If no errors occur during the procedure, when all the macros have been added to NEWMAC MACLIB, NEWMAC MACLIB is renamed *libname* MACLIB. The current *libname* MACLIB, if it exists, is erased.

   If errors occur during the VMFMAC EXEC procedure (for example, if a MACRO or COPY file is not found), *libname* MACLIB is not erased; and the updated macro library retains the name NEWMAC MACLIB.

Responses

Disk Input Files

***libname* EXEC**
List of MACRO or COPY files (or both) to be updated or included (or both) in *libname* MACLIB

***ctlfile* CNTRL**
Control file used by the UPDATE command (optional)

Input can also include MACRO and COPY files to be updated or included (or both) in the macro library, plus miscellaneous auxiliary control files (*fn* AUX*xxxxx*) and update files.

Disk Output Files

***libname* MACLIB**
Updated macro library

***libname* COPY**
Contains the UPDATE files produced by UPDATE command processing.

Note: This output file is created only if you specify a control file on VMFMAC.

Printer Output File

The printer is spooled with the CONT option. When VMFMAC completes, the printer file contains a copy of each macro or copy file in the macro library.

If you specify a control file, the printer file also contains:

- A copy of the control file
- For each updated macro or copy file, the update log file produced by the UPDATE command
- The *libname* COPY file, which contains the accumulated UPDATES files the UPDATE command created.

```
fn COPY|MACRO ADDED
```

The specified MACRO or COPY file has been added to the macro library.

```
libname COPY ADDED
```

The *libname* COPY file, containing the update history of the MACLIB, has been added.

```
*** TYPE 'VMFMAC LIBNAME <CTL>' ***
```

The VMFMAC command was entered with no operand(s).

```
*** libname EXEC NOT FOUND ***
```

VMFMAC could not locate the EXEC file associated with the macro library.

```
*** ctlfile CNTRL NOT FOUND ***
```

VMFMAC could not locate the specified control file.

```
*** fn COPY|MACRO NOT FOUND ***
```

The *fn* member named in *libname* EXEC could not be located.

```
*** ERRORS UPDATING fn COPY|MACRO ***
fn COPY|MACRO NOT INCLUDED IN MACLIB
```

An UPDATE command error occurred for library member *fn*, and the file was not written into the MACLIB.

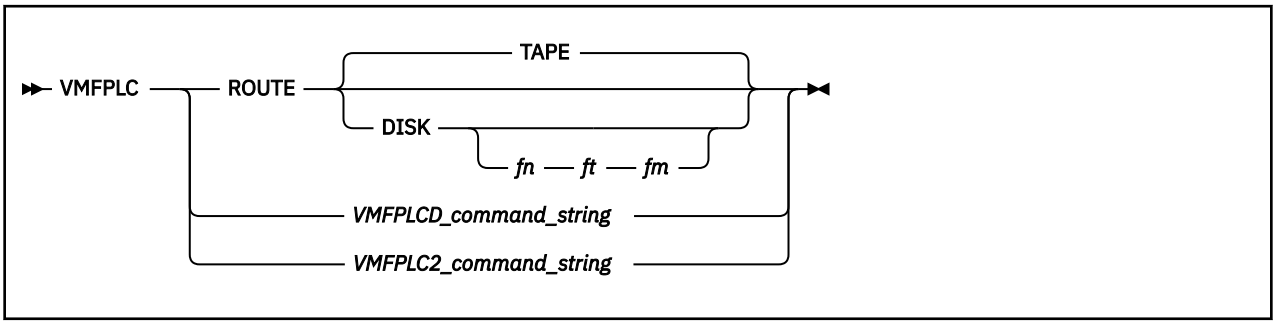
```
DUE TO PREVIOUS ERRORS, THE RESULT OF THIS MACLIB BUILD
IS CALLED 'NEWMAC MACLIB', libname MACLIB HAS
NOT BEEN REPLACED
```

One or more errors were encountered, and you must correct them and create the MACLIB yourself.

Messages and Return Codes

- DMS178I {Applying|Updating} *fn ft fm*
- DMS181E No update files were found [RC=40]

VMFPLC



Authorization

Service Installer

Purpose

The VMFPLC command is a common exec that calls either VMFPLC2 or VMFPLCD. VMFPLC supports the VMFPLC2 and VMFPLCD command formats. A ROUTE function allows VMFPLC to establish command routing for subsequent VMFPLC2 or VMFPLCD commands issued by VMFPLC.

VMFPLC acts as a front-end processor to minimize changes to existing routines that use VMFPLC2 when you want to convert to VMFPLCD or have a dual path. VMFPLC allows you to use either command format, including a mixture of the two sets of functions, parameters, and options.

Operands

ROUTE
establishes the routing for subsequent commands issued to VMFPLC which are to be passed to VMFPLC2 or VMFPLCD, or to reset any previous routing established. ROUTE is not passed to VMFPLC2 or VMFPLCD.

If ROUTE is not specified, it is assumed you are entering a VMFPLC2 or VMFPLCD command string.

If you do not specify DISK or TAPE with the ROUTE operand, ROUTE resets the routing variable to TAPE, regardless of any previous setting.

Options

DISK
establishes routing to disk (VMFPLCD). If the routing is to disk, the file identifier of the envelope may also be established (optional). If you specify an envelope file identifier, you must specify the complete file ID, *fn ft fm*. The file ID is checked for validity only to the extent that there are three parameters.

fn
is the file name of the file or envelope.

ft
is the file type of the file or envelope.

fm
is the file mode of the file or envelope.

TAPE
establishes routing to tape (VMFPLC2).

Usage Notes

1. When you issue VMFPLC, and not VMFPLC2 or VMFPLCD directly, VMFPLC routes to VMFPLC2 or VMFPLCD according to the routing in the VMFPLC GLOBALV variable from a VMFPLC ROUTE command. If no routing is established, the default routing is VMFPLC2.
2. Once the command routing has been determined, the options and control functions are converted (if necessary) to those appropriate for the chosen routing as follows:

   a. Options WTM, NOWTM, EOT, and EOF from VMFPLC2, and WGS, NOWGS, EOD, and EOG from VMFPLCD are converted to the respective keyword for the command to be called.

   VMFPLC2 options TAP*n*, VDEV, DEN *den*, and *nn*TRACK are deleted if the routing is to VMFPLCD.

   b. The control functions are converted as follows:

   VMFPLC2
   VMFPLCD

   MODESET
   No equivalent (See Note 1)

   ERG
   No equivalent (See Note 1)

   B/FSR
   No equivalent (See Note 2)

   B/FSF
   B/FSG

   REW
   RST

   RUN
   RST (See Note 3)

   WTM
   WGS

   Note:

   i) VMFPLC2 MODESET and ERG functions have no equivalent function in VMFPLCD and are handled as a 'no operation' request by VMFPLC if the command is for disk. For example, a return code of zero will be passed back without calling VMFPLCD.

   ii) VMFPLC2 BSR and FSR have no equivalent function in VMFPLCD. The functions are passed unchanged to VMFPLCD (if the medium is DISK) which results in an error because these functions are supposed to alter the position within the tape or envelope and no position change occurs.

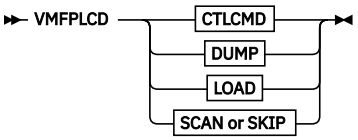
   iii) VMFPLC2 REW and RUN functions both convert to a VMFPLCD RST function. The VMFPLCD RST function converts to the VMFPLC2 REW function, **not** the RUN function.
3. VMFPLC does not check command syntax or option validity. With the exception of the previous keyword/command handling, the command is passed intact. Thus, all errors are issued by VMFPLC2 or VMFPLCD.
4. If the envelope file ID is provided with the DISK option, it is used to set or reset the GLOBALV variable used by VMFPLCD to remember the file ID across multiple invocations. If the file ID is different from that in an existing GLOBALV, or there is no existing GLOBALV, the logical record position in the envelope is set to record 1. If the GLOBALV variable exists and the file ID on the ROUTE function is the same as in the GLOBALV, no change in the logical record pointer is made.
5. Routing information is maintained in a GLOBALV variable for VMFPLC across multiple invocations of VMFPLC. The GLOBALV variable is maintained only within a CMS IPL; it is reset automatically on IPL. If ROUTE is issued with no parameters, the VMFPLC GLOBALV routing variable is deleted. This does not reset the file ID for the envelope file used by VMFPLCD.

Messages and Return Codes

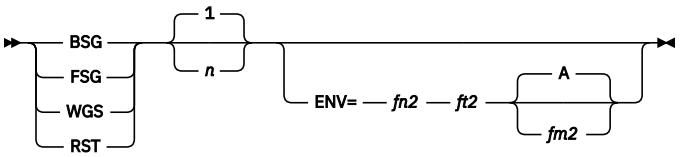
- DMS0003E Invalid option *option* [RC=24]
- DMS1447E Invalid command format [RC=24]

Note: Any other messages you may receive are returned from the VMFPLC2 or VMFPLCD commands. For more information on these messages, see "VMFPLCD" on page 1123 and "VMFPLC2" on page 1131.

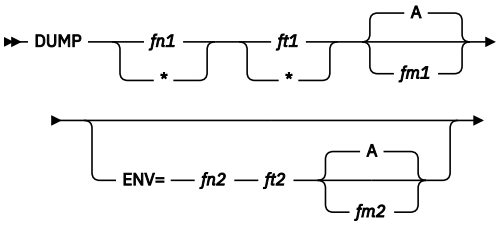
VMFPLCD

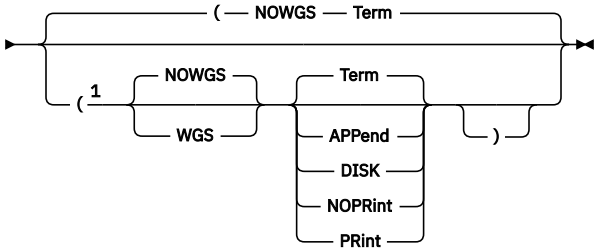


CTLCMD

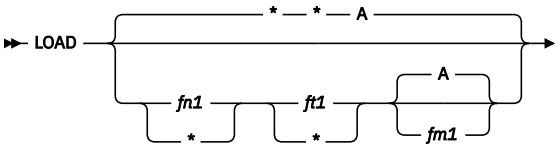


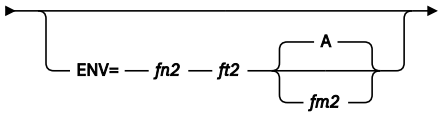
DUMP

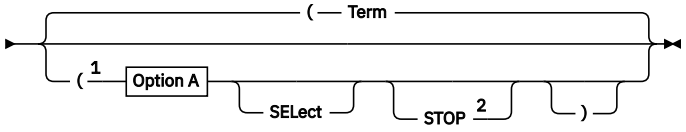




LOAD





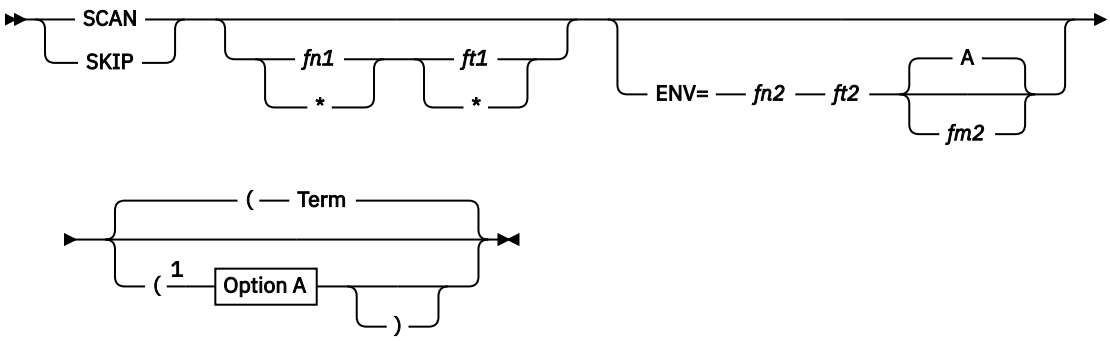


Notes:

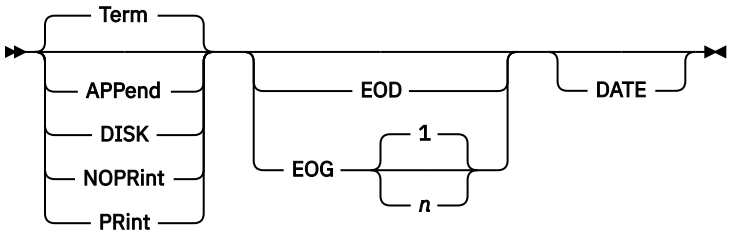
[1] You can enter Options in any order between the parentheses.

[2] The STOP option is ignored if you specify *fn1* or *ft1* with wild card substitution characters.

SCAN or SKIP



Option A



Notes:

[1] You can enter Options in any order between the parentheses.

Authorization

Service Installer

Purpose

Use the VMFPLCD command to:

- Load files from the envelope file
- Dump CMS files to the envelope file (files to be dumped can contain either fixed-length or variable-length records)
- Load previously dumped files from the envelope file
- Do various control operations on the envelope file

Note: Files processed by VMFPLCD must be in CMS format.

Operands

DUMP
dumps one or more CMS files to an envelope file.

The envelope file that is to receive the dumped file is identified by the ENV=*fn2 ft2 fm2* parameters. For more information on the default assignment if the ENV= parameter is not supplied, see Usage Note “3” on page 1128. The *fm2* must be for a disk accessed in write mode.

Note: The envelope file may not be dumped to itself. For more information, see Usage Note “13” on page 1129.

fn1
is the file name of the CMS file to dump.

*****
indicates all files with the file type and file mode specified. For more information, see Usage Note "13" on page 1129.

ft1
is the file type of the CMS file to dump.

*****
indicates all files with the file name and file mode specified. For more information, see Usage Note "13" on page 1129.

A
is the default file mode.

fm1
is the file mode of the CMS file to dump.

If you specify *fm1* as a letter, that file mode and its extensions are searched for the specified files. If you specify *fm1* as a letter and number, only files with that file mode number on that file mode and its extensions are dumped. If you do not specify the *fm1*, only file mode A and its extensions are searched.

*****
indicates all files with the file name and file type specified on all accessed disks. If you use wild card substitution, specifying *fm1* as an asterisk (*), all file modes are searched for the specified files. For more information, see Usage Note "13" on page 1129.

LOAD
copies the specified files, which are contained in an envelope file created by DUMP, into separate disk files with the file identifier of the original files. The envelope file ID is identified by the ENV=*fn2 ft2 fm2* parameter. For more information on the default assignment if the ENV= parameter is not specified, see Usage Note "3" on page 1128.

Note: No file may be loaded from the envelope that would overlay the envelope. For more information, see "Usage Notes" on page 1128.

fn1
is the file name of the file to copy.

For more information on specifying *fn1* with the wild card characters, see "Usage Notes" on page 1128.

*****
copies all files with the specified file type and file mode.

ft1
is the file of the file to copy.

For more information on specifying *ft1* with the wild card characters, see "Usage Notes" on page 1128.

*****
copies all files with the specified file name and file mode.

A
is the default file mode.

fm1
is the file mode of the file to copy. The files are written to the disk identified by the file mode letter. *fm1* must be a disk accessed in read/write (R/W) mode. When a file mode is specified, only files with that file mode number are loaded.

*****
copies all files with the specified file name and file type.

The *fn1* and *ft1* may be specified using wild card substitution characters. If wild card substitution is used, any file in the envelope that satisfies the specified *fn1 ft1* are loaded within the limits set by

options EOG *n* or EOD. For more information on wild card specification, see "Usage Notes" on page 1128.

EOG or EOD controls the limit of the search in the envelope file for a disk file to be loaded.

If you specify a file identifier (*fn1 ft1*) for the file to be loaded, only that one file is copied from the envelope file. If no file identifier is specified, all files within the *n* file groups from the option EOG *n* (or in the entire envelope file for option EOD) are copied.

ENV=
represents the envelope file to be processed.

Note: The ENV keyword must always be followed by a blank. For more information on the default assignment if the ENV= parameter is not specified, see Usage Note "3" on page 1128.

fn2
is the file name of the envelope to be processed.

ft2
is the file type of the envelope to be processed.

A
is the default file mode.

fm2
is the file mode of the envelope file to be processed. *fm2* must be a disk accessed in write mode.

SCAN
positions the envelope file at a specified point and lists the identifiers of the files within the envelope file scanned. Scanning occurs over *n* file groups, as specified by the option EOG *n*, the default is 1 or over the entire envelope file (EOD option).

If you specify a specific disk file identifier (*fn1 ft1*), scanning stops when that file is encountered. The position within the envelope file is set to the header record of the file that satisfies the file identifier.

fn1
is the file name of the file to scan for.

***
scans all files that match the specified file type (*ft1*).

ft1
is the file type of the file to scan for.

***
scans all files that match the specified file name (*fn1*).

The *fn1* and *ft1* may be specified using wild card substitution characters. For more information, see "Usage Notes" on page 1128.

If wild card substitution symbols are used, the file identifier is not used to locate a specific file on which to stop the scan for positioning. Instead, the file identifier is used only to determine which files will be logged.

If *fn1* and *ft1* are not provided and logging is requested, all files encountered are logged.

SKIP
positions the envelope file at a specified point and lists the identifiers of the files within the envelope file skipped.

fn1
is the file name of the envelope file to skip.

***
skips all files that match the specified file type (*ft1*).

ft1
is the file type of the envelope file to skip.

*****
skips all files that match the specified file name (*fn1*).

The *fn1* and *ft1* may be specified using wild card substitution characters. If wild card substitution symbols are used, the file identifier is not used to locate a specific file on which to stop the scan for positioning. Instead the file identifier is used only to determine which files will be logged. For more information on wild card substitution, see "Usage Notes" on page 1128.

If a specific disk file identifier (*fn1 ft1*) is specified, skipping stops when that file is encountered. The position within the envelope file is set to the header record of the next file, or the group separator record, whichever follows.

If *fn1* and *ft1* are not provided and logging is requested, all files encountered are logged.

Skipping occurs over *n* file groups, as specified by the option EOG *n*, the default is 1 file group or over the entire envelope file (EOD option).

BSG
Backspaces *n* file group separators.

FSG
Forward spaces *n* file group separators.

WGS
Writes *n* file group separators.

RST
Resets logical position to start of the file envelope file, and resets the file ID of the envelope file.

1
is the default number of group separators.

n
is the number of file group separators.

Options

NOWGS
indicates no group separator is written after each file is dumped. The default is NOWGS.

WGS
writes a file group separator after each file is dumped.

Term
displays a list of files dumped, loaded, scanned, or skipped at the terminal. The default is term.

APPend
causes the disk file containing the list of files dumped, loaded, scanned, or skipped to be added to the end of an existing list file (DISK MAP A5).

DISK
creates a CMS file called DISK MAP A5 containing the list of files dumped, loaded, scanned, or skipped.

NOPRint
does not list the files dumped, loaded, scanned, or skipped.

PRint
spools the list of files dumped, loaded, scanned, or skipped to the printer.

EOD
reads the envelope file until the end of disk condition is recognized. End of disk is signaled by either two successive file group separators or an end of file condition during a read of the envelope file.

EOG
reads the envelope file through a maximum of *n* file group separators (end of group records).

1
is the default.

n
is the number of group separators.

SELect
inhibits loading of a file from the envelope file that causes replacement of an identical file in virtual direct access storage. Files will be loaded only if they do not exist on the disk specified by the LOAD command, or when the date/time stamp for the file being loaded **does not match** the date/time stamp for the existing file.

STOP
assumes files contained in the envelope file are in alphabetic sequence by file name and file type. If the requested file is found, the file is loaded; and the envelope file is positioned at the next record following the loaded file. If the file is not in the envelope and a file is encountered that is alphabetically beyond the bounds of the requested file, no file is loaded and the envelope file is positioned to load this file. You must specify the disk file *fn1* and *ft1* without using wild card substitution characters. If either the *fn1* or the *ft1* contains substitution characters, the STOP option will be ignored.

DATE
displays LISTFILE information during a load, scan, or skip function. The record format, logical record size, number of records, and date/time stamp are displayed.

Usage Notes

1. VMFPLCD is restricted to a maximum LRECL value of 65535 when dumping CMS files to an envelope file.
2. All control functions, except RST, allow you to specify a repetition factor (*n*) to control the number of groups skipped or the number of group separator records to be written.
3. The file ID of the envelope file is controlled by two separate factors: a GLOBALV variable saved between VMFPLCD invocations, and the ENV= parameter.
   - If the ENV= parameter is provided, it specifies a file ID for the envelope file which takes precedence over the other methods of specification. In this case, the *fn2* and *ft2* must be provided (wild card substitution is not allowed). The *fm2* is optional, and the default file mode is A.
   - If the ENV= parameter is not specified, the GLOBALV variable is checked to find the file ID of the envelope file. If present, the GLOBALV variable for the file ID will be used. This allows the first invocation of the command to set the file ID and all subsequent commands for the same envelope file to be entered without the ENV= parameter.

     Note: All functions except RST, DUMP, and WGS require the envelope file to exist. If it does not, the function fails and the GLOBALV variable is not updated with the envelope file ID.
   - If there is no ENV= parameter and no GLOBALV variable, an error is returned.
4. The GLOBALV variable is maintained for one envelope file at a time, to remember the file ID and the logical position across multiple invocations of VMFPLCD. When a new envelope file ID is used, the variable is automatically reset to the logical start of the envelope file. The variable is maintained only within a CMS IPL. It is automatically reset to null on each CMS IPL.
5. If you choose to change the envelope file ID within the same IPL, use the ENV= parameter. Except for RST, DUMP, or WGS, the envelope file must exist. Otherwise, the exec will end with an error; and the GLOBALV variable will not be updated.
6. If the prior envelope file is erased and you issue a VMFPLCD command with no ENV= parameter, the RST, DUMP, or WGS commands establish a new envelope with the same name as the prior one.
7. If this exec issues an error with a return code other than 24, the GLOBALV variable is updated. On input/output errors, the position is set to the last known position prior to the error.
8. Because the end of disk condition is recognized as either two successive group separator records or the physical end of the envelope file, logical positioning functions that involve the end of disk condition are handled as if there were two group separator records at the end of the envelope file. This provides a consistent external result that matches VMFPLC2, regardless of what is actually

present. The actual end of the envelope file could have none, one, or two real group separators. If there are less than two real group separators, the required number of positions beyond the last record will be treated as group separators to satisfy this requirement.

9. BSG scans the envelope backward to locate a group separator record and leaves the position ready to read this group separator. FSG scans forward to locate a group separator and leaves the position ready to read the following record.
10. If you choose to reset the logical position to the start of the envelope file, use the RST control function.
11. The LOAD, SKIP, and SCAN functions leave the position after the file or the group separator that satisfied the conditions specified. SCAN can also leave the position ready to read a specific file.
12. DUMP and WGS leave the position ready to write at the next record position after the dumped files or group separator records written as a result of the command. If the logical position at the start of a DUMP or WGS separator is other than the next record past the actual physical end of the envelope, the envelope is adjusted before the function, to either truncate it or convert the missing group separator records to real records.
13. When specifying the *fn1 ft1* or *fm1* identifiers for the disk files, special wild card characters may be used, where indicated in the option and operand descriptions, to allow any characters to satisfy all or part of the identifier. The character * represents any number of characters including zero. As many asterisks as required can appear anywhere in a file name or file type. Only one asterisk can be used for file mode. The character % 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. The percent symbol may not be used for file mode.

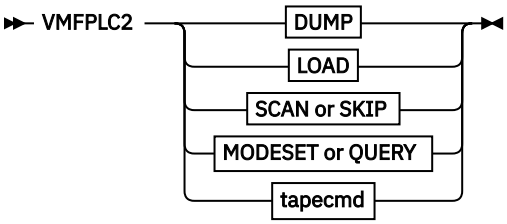
    When wild card substitution is used during a DUMP function to select the files to dump, the order in which the files are dumped is the same order in which the files would be found with a CMS LISTFILE command. This order can vary from session to session. Thus, if a specific order is needed, do not use wild card substitution.
14. The *fn ft fm* parameters (disk file ID or envelope file ID) must conform to CMS rules. For example, the file name and file type must be eight positions and the file mode is two positions. If they are longer, they are truncated to the maximum length and no error message is issued.
15. Conflicting options (for example, WGS and NOWGS) result in an error from VMFPLCD. This is not treated as an error with VMFPLC2. Instead, the last value entered is used.
16. VMFPLCD dumps files that have a *fn1* or *ft1* that contains mixed-case characters only if wildcard symbols are used in the file ID. For example, the file Test FILE can be dumped with a file ID of T%%% FILE, * FILE, T* *, or other such IDs, but will fail if requested as Test FILE.
17. If an attempt is made to dump the envelope file (or a copy of it) to itself, or if a file to be loaded from the envelope would overlay the envelope, processing is stopped; and an error occurs.
18. An envelope may contain other envelope files, but not itself. All control records within the envelope contain a prefix section that identifies the envelope. This prefix is established on the first output function performed and remains the same even if the envelope file is renamed in the middle of processing. Two different envelope files that were created with the same *fn* and *ft* will appear to VMFPLCD as the same envelope even if one is renamed after creation.
19. The envelope file created by VMFPLCD is not packed to condense the size. Packing of the envelope prior to sending it to another user will greatly reduce the size of the file. Before processing the envelope again, it must be unpacked.

Messages and Return Codes

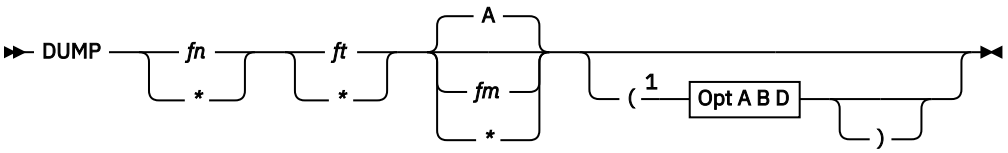
- DMS0002E File VMFPLCM EXEC * not found [RC=28]
- DMS0002E File *fn ft fm* not found [RC=20|28|36]
- DMS0003E Invalid option: *option* [RC=24]
- DMS0010S Premature end occurred on *fn ft fm* [RC=40]
- DMS0014E Invalid function *function* [RC=24]

- DMS0029E Invalid parameter *parameter* in the option *option* field [RC=24]
- DMS0040E No files loaded [RC=28]
- DMS0048E Invalid filemode *filemode* [RC=24]
- DMS0066E *option option option option option* are conflicting options [RC=24]
- DMS0069E Filemode *filemode* not accessed [RC=36]
- DMS0104S Error *rc* reading file *fn ft fm* from disk or directory [RC=100]
- DMS0105S Error *rc* writing file *fn ft fm* on disk or directory [RC=100]
- DMS0671E Error loading file *fn ft fm*; rc=*rc* from COPYFILE [RC=100]
- DMS0814E Message number *num*, format *fmt* was not found [RC=12]
- DMS0814E Dictionary number *num* format *fmt* not found [RC=12]
- DMS1447E Invalid command format [RC=24]
- DMS1448E Option(s) *option option option option option option option option option option option option* invalid for *function* [RC=24]
- DMS1450S Output file *fn ft fm* disk is read-only [RC=36]
- DMS1451S Cannot dump an envelope file to itself. File *fn ft fm* is same envelope file [RC=32]
- DMS1452S Unidentifiable envelope control record in *fn ft fm* [RC=100]
- DMS1453S Loading of file *fn ft fm* would overlay envelope [RC=32]
- DMS1454S Date/time stamp update on *fn ft fm* failed [RC=104]
- DMS1455S Error occurred trying to truncate file *fn ft fm* [RC=100]
- DMS1456S Unexpected error *rexx code* from VMFPLCD [RC=104]
- DMS1457E Envelope file was not specified [RC=24]
- DMS1458E File designated as envelope on *function* is not an envelope [RC=32]
- DMS1459S Number of records for file *fn ft fm* different than when dumped [RC=40]
- DMS1460E Envelope file *fn ft* exists but is on R/O extension of *fm* disk [RC=36]
- DMS1498E Envelope file *fn ft* is not in PLCD format [RC=100]
- DMS1499E Mixed envelope format [RC=100]

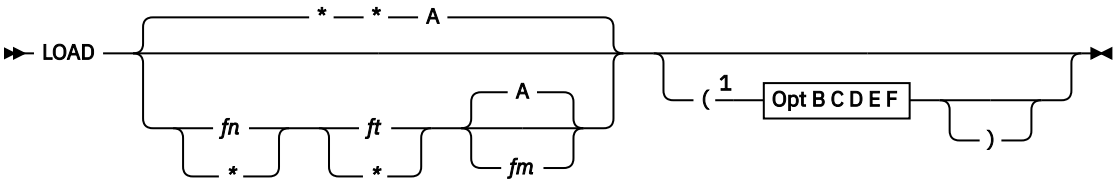
VMFPLC2



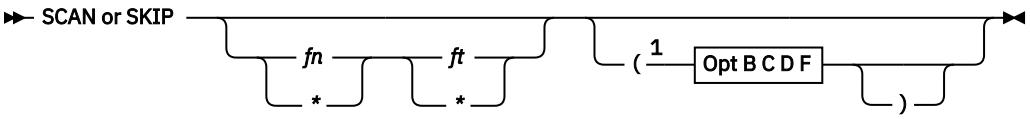
DUMP



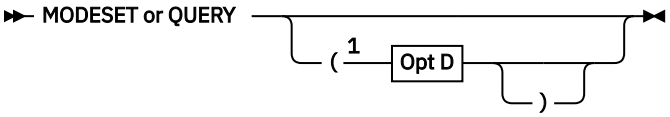
LOAD



SCAN or SKIP



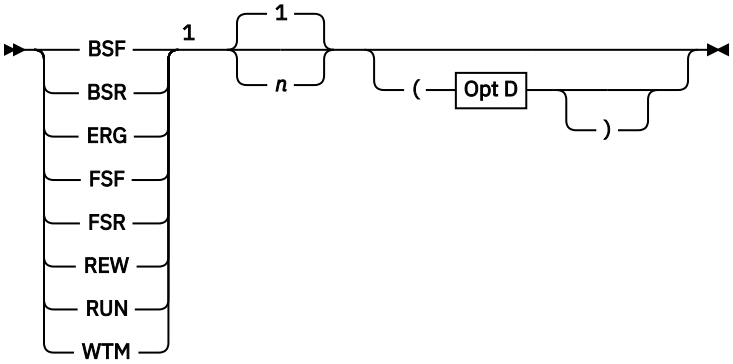
MODESET or QUERY



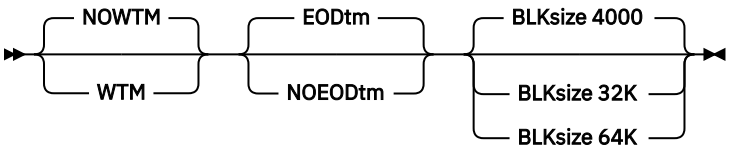
Notes:

[1] You can enter Options in any order between the parentheses. If conflicting options are specified, an error message is issued.

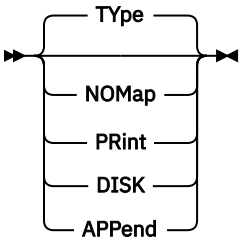
tapecmd



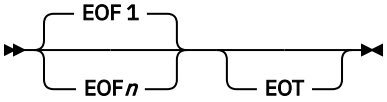
Opt A



Opt B



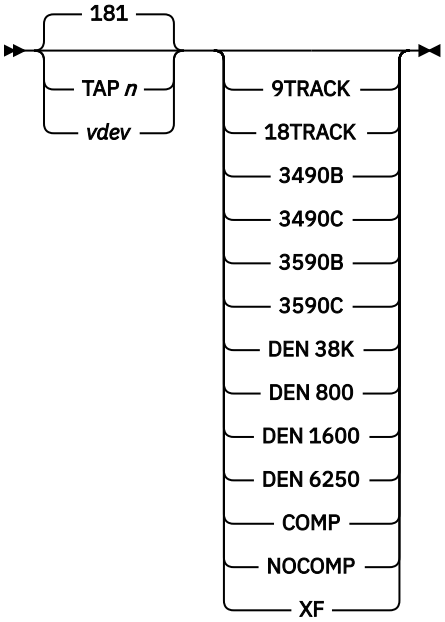
Opt C



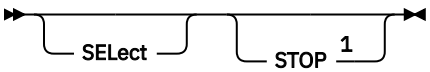
Notes:

[1] ERG, REW, and RUN do not require a numeric value.

Opt D



Opt E



Opt F

Notes:

[1] If *fn* or *ft* is specified as *, STOP is ignored.

Authorization

General User

Purpose

Use the VMFPLC2 command to:

- Load files from the z/VM product tape and z/VM service tapes
- Dump CMS files to tape (files to be dumped can contain either fixed-length or variable-length records)
- Load previously dumped files from tape
- Do various control operations on a specified tape drive

The VMFPLC2 command does not process multivolume files. Files processed by the VMFPLC2 command must be CMS-formatted.

For more information on working with tapes, see "TAPE" on page 1061.

Operands

DUMP
dumps one or more CMS files to tape. If you do not specify the file mode, the default is A.

If *fn ft* are fully specified (such as MY FILE), extensions of your A disk or directory will not be picked up.

If you specify file name or file type, or both, as an asterisk (*), all files that satisfy the other file identifier(s) are dumped, including files on extensions of the file mode specified (or the default of A). If you specify file mode as an asterisk (*), all file modes are searched for the specified file(s).

LOAD
reads tape files into a minidisk or directory. If you specify a file identifier with no asterisks, that one file is loaded. If you specify the end of file (EOF) *n* option without a file identifier, *n* tape files are loaded.

Note: The *tape file* consists of all CMS files between two tape marks. If you specify *fn* or *ft* as an asterisk (*), all files within EOF *n* that satisfy the other file identifier are loaded. You must specify both identifiers.

The files are written to the specified file mode. If you do not specify a file mode, the default is A. If you also specify a file mode number, only files with that file mode number are loaded. If you do not specify *fn*, *ft*, or *fm*, all files up to EOF or end of tape are loaded to your disk or directory accessed as A.

SCAN
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 you specify *fn* and *ft*, scanning stops upon encountering that file, with the tape positioned to load the specified file.

You must be positioned after the volume label (if one exists) to use SCAN.

SKIP
positions the tape at a specified point and lists the identifier of the file it skips. Skipping occurs over *n* tape marks, as specified by the option EOF *n* (the default is 1 tape mark). However, if you specify *fn* and *ft*, skipping stops after encountering that file, with the tape positioned immediately following the file.

QUERY
displays information about a tape device, particularly which recording formats the device is capable of writing.

In response to the QUERY option, CMS issues the following message for each of the recording formats which the device is capable of writing:

```
DMS217R  Device vdev can write recording format (option)
```

MODESET
does nothing except set default recording format options, which is done with all subcommands. For more information on setting default recording format options, see the TAPE command in *z/VM: CMS User's Guide*.

tapcmd n
specifies a tape control function. The valid tape control functions are shown in the following table. For the BSF, BSR, FSF, FSR, and WTM functions the number of executions can be specified in the command. If you do not specify the number of executions, the default is 1. If a 0 is specified, the function is ignored. These functions also work on tapes in a non-CMS format.

tapcmd
Action

BSF *n*
Backspace *n* tape marks

BSR *n*
Backspace *n* tape records

ERG
Erase gap (defective section)

FSF *n*
Forward-space *n* tape marks

FSR *n*
Forward-space *n* tape records

REW
Rewind tape to load point

RUN
Rewind tape and unload

WTM *n*
Write *n* tape marks

If you specify EOF *n*, FSF *n*, or FSR *n*, where *n* is greater than the number of tape marks or tape records on the tape, the tape will run off the reel. If a BSF *n* or BSR *n* command causes the tape to return to loadpoint. For example, if you enter:

```
vmfplc2 bsf 3
```

when there is only one tape mark to backspace over), VMFPLC2 issues the DMS110S message and processing stops.

Options

Note: If conflicting options are specified, an error message will be issued.

WTM
writes a tape mark after each file dumped.

NOWTM
does not write a tape mark after each file dumped. The default is NOWTM.

EODtm
writes two tape marks after the last file dumped. (Two tape marks indicate the end of tape (EOT).) It then backspaces over one or two tape marks to position the tape in case more files are to be dumped. The default is EODTM.

NOEODtm
writes no additional tape marks at the end of the last file dumped.

Attention: If two tape marks do not follow the last file on a tape, some tapes like the 3480 can give unpredictable results. If you specify NOEODTM, you should ensure there are at least two tape marks at the end of your *final* dump by either using the default EODTM or issuing `TAPE WTM 2` after your final dump.

Table 51 on page 1135 shows the results of specifying various combinations of EODTM, NOEODTM, WTM, and NOWTM:

Table 51. Results of EODTM, NOEODTM, WTM, and NOWTM Options Combinations

Options Specified	Tape Marks Between Files	Tape Marks at End of Files	Tape is Positioned
WTM WTM EODTM	1	2	Between last 2 tape marks
WTM NOEODTM	1	1	After last tape mark
NOWTM NOWTM EODTM EODTM	0	2	Before last 2 tape marks
NOWTM NOEODTM NOEODTM	0	0	At end of files

BLKsize
specifies the approximate size of the tape block to which the files are dumped. The larger the block size, the more data will fit on a tape.

The data capacity of VMFPLC2 tape depends on the BLKSIZE option used:

BLKSIZE	Capacity
4000	4000 bytes of data, plus control information
32K	32,767 (32K-1) bytes of data, plus control information
64K	65,535 (64K-1) bytes of data, and control information

The default is BLKSIZE 4000.

NOMap
suppresses all MAP output. The list of files processed is not spooled to any output device (console, printer, or disk), and no TAPE MAP file is created.

Note: The NOMap option replaces the NOPRint option. NOPRint is still functional, but is no longer supported.

PRint
spools the list of files dumped, loaded, scanned, or skipped to the user's virtual printer.

TYpe
displays a list of files dumped, loaded, scanned, or skipped. The default is type.

Note: The TYPE option replaces the TERM option. TERM is still functional, but is no longer supported.

DISK
creates a CMS file called TAPE MAP A5 containing the list of files dumped, loaded, scanned, or skipped.

APPend
adds the TAPE MAP file (containing the list of files dumped, loaded, scanned, or skipped) to the end of an existing TAPE MAP file.

EOF *n*
reads the tape through a maximum of *n* tape marks. If the EOF option is not specified, the default is EOF 1. If EOF *n* is specified, you must specify a number for *n*, or you will receive error DMS0393E.

EOT
reads the tape until an end of tape indication (two tape marks) is recognized. You may want to specify EOT after EOF *n* to prevent going past the end of tape if *n* is greater than the actual number of tape marks.

DATE
displays listfile information during a SCAN, SKIP, or LOAD. The information displayed includes number of records, length of records, and date/time stamp.

TAP*n*
vdev
specifies the device name (TAP*n*), or alternatively the virtual device number (*vdev*) of the tape device on which the command is to operate. For more information on the device names and virtual device numbers for tape devices, see *z/VM: CMS User's Guide*. These names and corresponding virtual device numbers are valid:

Device Name	Device Number	Device Name	Device Number
TAP0	0180	TAP8	0288
TAP1	0181	TAP9	0289
TAP2	0182	TAPA	028A
TAP3	0183	TAPB	028B
TAP4	0184	TAPC	028C
TAP5	0185	TAPD	028D
TAP6	0186	TAPE	028E
TAP7	0187	TAPF	028F

You may omit the leading zero on the device numbers. The default is TAP1.

3490C
specifies 3490 Compacted recording format.

3490B
specifies 3490 Basic recording format.

18TRACK
specifies 3480 Basic recording format.

XF
specifies 3480 Compacted recording format.

3590C
specifies 3590 Compacted recording format.

3590B
specifies 3590 Basic recording format.

DEN 6250
specifies GCR recording format.

DEN 1600
specifies PE recording format.

DEN 800
specifies NRZI recording format.

DEN 38K
specifies the 3480 Basic recording format.

DEN 38K is equivalent to 18TRACK, which is the preferred option.

9TRACK
specifies "any 9 track recording format." CMS selects a 9 track recording format for you that the device is capable of writing (if there is one). If you require a particular 9 track recording format, use the DEN 800, DEN 1600, or DEN 6250 option.

COMP
Specifies "any compacted recording format." CMS selects a compacted recording format for you that the device is capable of writing (if there is one). If there is a choice of formats, CMS will choose the one which will fit the greatest amount of data on the tape. If you require a particular compacted recording format, use the 3590C, 3490C, or XF option.

NOCOMP
Specifies "any uncompacted recording format." CMS selects an uncompacted recording format that the device is capable of writing (if there is one). If there is a choice of formats, CMS will choose the one which will fit the greatest amount of data on the tape. If you require a particular uncompacted recording format, use the 3590B, 3490B, 18TRACK, DEN 6250, DEN 1600, or DEN 800 option.

SELect
inhibits loading of a file from the tape that causes replacement of an *identical* file in virtual direct access storage. If a file already exists on the specified file mode with the same date/time stamp as the file on the tape, the file will not be loaded, and no warning message will be issued.

STOP
assumes the files contained on the tape are in alphabetic sequence. If the requested file is on the tape, the file is loaded and the tape stops. If the file is not on the tape, the tape will stop when a file is encountered whose name comes later in the alphabet than the requested file. If an asterisk is specified for either the file name or file type, the STOP option is ignored.

When the STOP option is used with the LOAD subcommand, and the file does not exist, a return code of 44 is issued with message DMS002E.

Usage Notes

1. A corrupt file (for example, one having an incorrect FST) may not be detected.
2. If the tape is not correctly positioned (at the beginning of a file) at the start of an operation, this may not be detected and a corrupt file will result.
3. The first time VMFPLC2 is issued, module DMSP2C module is loaded as a nucleus extension. It remains there until NUCXDROP, IPL, or LOGOFF is issued.
4. If VMFPLC2 is defined as a synonym of the TAPE command (or the TAPE command is defined as a synonym of VMFPLC2), you cannot use the synonym to call the function from within an exec. You can use any name other than VMFPLC2 or TAPE as a synonym of the other function. For example, from within an exec TAPE is not a valid synonym for VMFPLC2; however, TAP is a valid synonym.
5. If there is not enough space on a file mode to hold the files contained in the tape file being loaded, load operations will terminate. To prevent this, when you dump the files, separate CMS (logical) files by tape marks, then forward space to the appropriate CMS file.
6. For more information about tape file handling, see *z/VM: CMS Application Development Guide for Assembler*.
7. If you try to load an empty file into a minidisk, or into a directory in a file pool that does not support empty files, no file is created and you receive warning message DMS636W. Processing will continue.
8. For more examples on using the VMFPLC2 command, see *z/VM: CMS User's Guide*.

9. If the tape is under the control of a Tape Library Dataserver machine, and the DFSMS/VM Removable Media Services (RMS) FSMPPSI CSLLIB is available to CMS, the RUN function calls the RMS FSMRMDMT (Demount) CSL routine to have the Dataserver unmount the tape.

Responses

If the TYPE 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
   .  .  .
   .  .  .
   .  .  .
```

Messages and Return Codes

- DMS002E File *fn ft fm* not found [RC=28]
- DMS003E Invalid option: *option* [RC=24]
- DMSO03E Invalid option *option* with *function* function [RC=24]
- DMS010E Premature EOF on file *fn ft* [RC=40]
- DMS014E Invalid function *function* [RC=24]
- DMS023E No filetype specified [RC=24]
- DMS027E Invalid device *vdev* [RC=24]
- DMS029E Invalid parameter *parameter* in the option *option* field [RC=24]
- DMS037E Filemode *filemode* is accessed as read/only [RC=36]
- DMS042E No fileid(s) specified [RC=24]
- DMS043E TAP*n*(*vdev*) is file protected [RC=36]
- DMS047E No function specified [RC=24]
- DMS048E Invalid filemode *mode* [RC=24]
- DMS057E Invalid record format [RC=32]
- DMS058E End-of-file or end-of-tape [RC=40] [RC=32]
- DMS065E *option* option specified twice [RC=24]
- DMS066E *option1* and *option2* are conflicting options [RC=24]
- DMS069E Filemode *filemode* not accessed [RC=36]
- DMS104S Error *rc* reading file *fn ft fm* from disk or directory [RC=31|55|70|76|99|100]
- DMS105S Error *rc* writing file *fn ft fm* on disk or directory [RC=31|55|70|76|99|100]

- DMS110S Error reading TAP*n*(*vdev*) [RC=100]
- DMS111S Error writing TAP*n*(*vdev*) [RC=100]
- DMS113S TAP*n*(*vdev*) not attached [RC=100]
- DMS115S Device *name* cannot write the *format* recording format [RC=88]
- DMS115S Device *name* cannot write any {9TRACK|compacted} recording formats [RC=88]
- DMS173W Empty output file *fn ft fm* not created
- DMS217R Device *vdev* can write *recording format (option)*
- DMS335W TAP*n*(*vdev*) has been rewound and unloaded by operator. Requested tape function may not have been executed. [RC=4]
- DMS0393E Missing *valuetype* for option *option* [RC=24]
- DMS613E TAPE/VMFPLC2 must be invoked as a nucleus extension [RC=40]
- DMS636W File *fn ft fm* is empty; minidisk does not support empty files
- DMS671E Error loading file *fn ft fm*; rc=*rc* from RENAME [RC=31|55|70|76|99|100]
- DMS671E Error loading file *fn ft fm*; rc=*rc* from COPYFILE [RC=31|55|70|76|99|100]
- DMS1262S Error *rc* opening file *fn ft fm* [RC=31|55|70|76|99|100]
- DMS1262S Error *rc* closing file *fn ft fm* [RC=31|55|70|76|99|100]
- DMS2139I VDEV *vdev* SENSE gives ERA/RAC=*rc*; cartridge may not be valid for I/O
- DMS2147W Library Dataserver DEMOUNT error CSLRC= *nnn*, CSLRS= *nnnnn*, FCTRC= *nnn*, FCTRS= *nnnnn*, a direct REWIND and UNLOAD will now be attempted

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411
Errors in the Shared File System	“File Pool Server Messages” on page 1414

VMFTXT

Authorization

Service Installer

Purpose

Use the VMFTXT command to rebuild a named text library (TXTLIB) file using a member list contained in an EXEC file with the same file name.

To locate the member files, VMFTXT searches all your file modes using the standard search order, A through Z. The member list can specify an optional file type for specific members.

Parameters

libname
is the file name of the text library (TXTLIB) file you want to update. It is also the file name of the EXEC file that contains the names of the library members. The recommended format of the EXEC file is as follows:

```
&TRACE OFF
*Optional comments can be included
 &1 &2 [&3] fn [ft] [(FILename [) ] ]
 &1 &2 [&3] fn [ft] [(FILename [) ] ]
    .
    .
    .
```

In each entry, *fn* and *ft* are the file name and file type of an object file you want to add to the library. Specifying the file type is optional.

If you specify *ft*, VMFTXT looks for the specific file.

If you do not specify *ft*, and you do not specify a control file on the command line, VMFTXT looks for a file type of TEXT.

If you do not specify *ft*, but you do specify a control file on the command line, VMFTXT searches for the specified member in the file types determined by the control file.

Each entry in the member list EXEC file can also specify an optional file name parameter (FILename) to be passed directly to the TXTLIB command. This parameter indicates the member name is to be taken from the file name and not from the CSECT name within the file.

ctlfile
is the file name of an optional control file VMFTXT uses to determine the file types of the object files added to the text library. The file type of the control file must be CNTRL.

This file is usually the same control file used with the VMFASM or UPDATE commands to apply updates to modules. The control file identifies the file type search order if you do not specify the file type in the member list.

VMFTXT Processing

For each entry in the member list that does not specify a file type:

- If you do not specify a control file name on the command line, the default object module file type is TEXT.
- If you do specify a control file name on the command line:
  1. VMFTXT searches the control file to determine the file type of the object module. The file types are based on the update level identifiers in the control file. (VMFASM uses these identifiers to assign file types to object decks.) Remember that updates are applied to source files from the bottom of the control file toward the top. Therefore, VMFTXT searches the control file from the top toward the bottom to locate the most recent update level.

     For example, assume the control file contains the following records:

     ```
     TEXT  MACS DMSMAC
     LOCAL FIX1
     SPEC  AUX1111
     PTF   P12567DS
     IBM1  AUXVM
     ```

     Then, for each entry in the member list, the VMFTXT search order is:

     – *fn* TXTLOCAL
     – *fn* TXTSPEC
     – *fn* TXTIBM1
     – *fn* TEXT

     Note: When determining the file types of object files to add to the library, VMFTXT ignores records that have an update level identifier of PTF and searches for the next level identifier.
  2. When VMFTXT locates a file, it adds it to VMFTXT TXTLIB, then continues processing the next entry in the member list. If there is no text file, VMFTXT displays a message and continues processing with the next entry in the member list.
  3. When the procedure successfully adds all the members, VMFTXT issues the commands:

     ```
     ERASE libname TXTLIB A
     RENAME VMFTXT TXTLIB A libname TXTLIB A
     ```

Results

Disk Input Files

***libname* TXTLIB**
is the text library file to be updated.

***libname* EXEC**
contains the file names, and optionally the file types, of the object files to be included in the library.

object modules
are the files to be included in the library, with file types specified in the member list, or with file types of TEXT or TXT*nnnnn* Where:

nnnnn
specifies the update level identifier obtained from a control file.

Work Files

These file identifiers are reserved for use by VMFTXT:

```
* $VMFTXT$
VMFTXT TEXT
VMFTXT TXTLIB
```

If you have any of these work files on file mode A when you enter the VMFTXT command, VMFTXT issues an error message and halts processing with a return code of 28.

If you are adding a member that has a file type other than TEXT, and you already have a TEXT file for that member on file mode A, VMFTXT uses the $VMFTXT$ file type to temporarily rename the file during processing. After processing is complete, VMFTXT renames the file back to a file type of TEXT. If processing is interrupted for any reason, you should enter:

rename * $vmftxt$ a = TEXT =

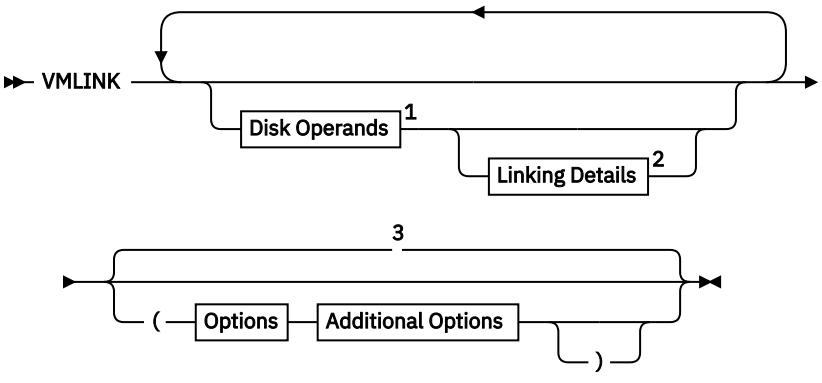
to rename the file.

Note: If you do not need the VMFTXT TEXT and VMFTXT TXTLIB files for problem diagnosis, you can erase them.

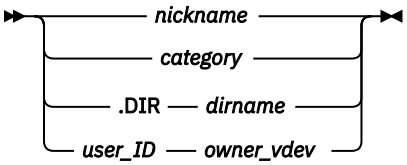
Messages

- DMS001E No filename specified [RC=24]
- DMS002E File *fn ft fm* not found [RC=28]
- DMS006E No read/write A filemode accessed [RC=36]
- DMS024E File *fn ft fm* already exists [RC=28]
- DMS026E Invalid parameter *parameter* for *function* function [RC=24]
- DMS056E File *fn ft fm* contains invalid record formats [RC=100]
- DMS062E Invalid character *character* in fileid *fn ft fm* [RC=100]
- DMS179E Missing or invalid MACS card in control file *fn ft fm*
- DMS183E Invalid CONTROL file control card [RC=32]
- DMS405E Invalid or missing message number [RC=24]
- DMX895I Member *fn ft* added
- DMS896E File *fn ft fm* not found
- DMS896E File *fn* TEXT or *fn* TXT*xxxxx* not found
- DMS897E Due to previous errors, the result of this TXTLIB build is called VMFTXT TXTLIB; your *fn* TXTLIB has not been replaced [RC=40]

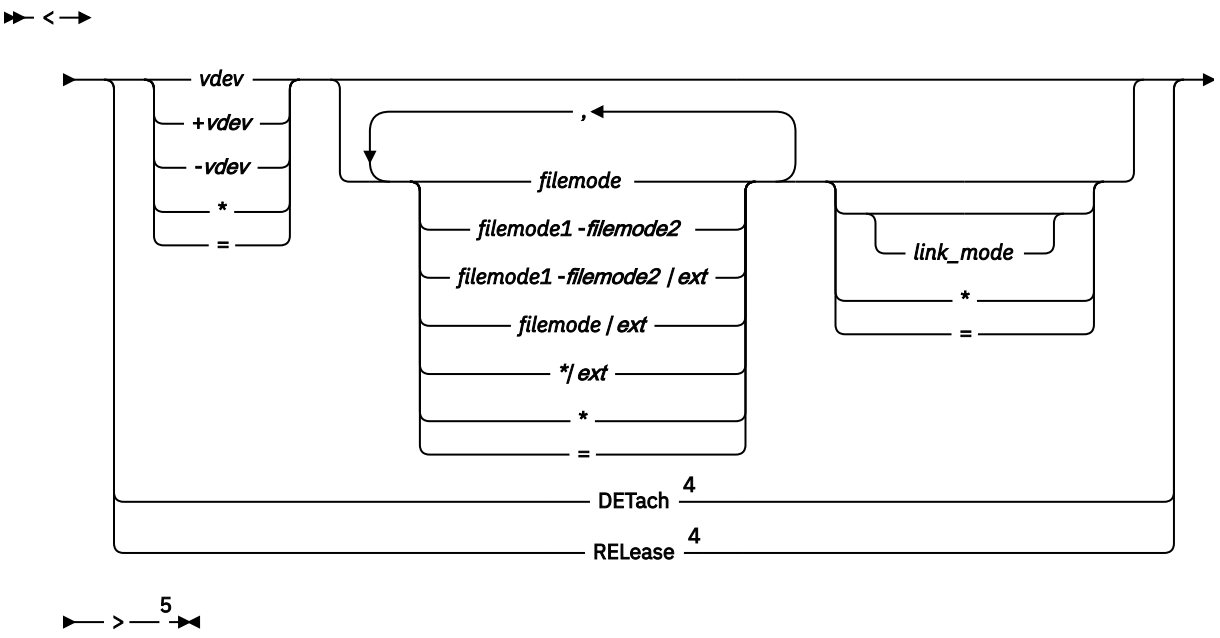
VMLINK



Disk Operands



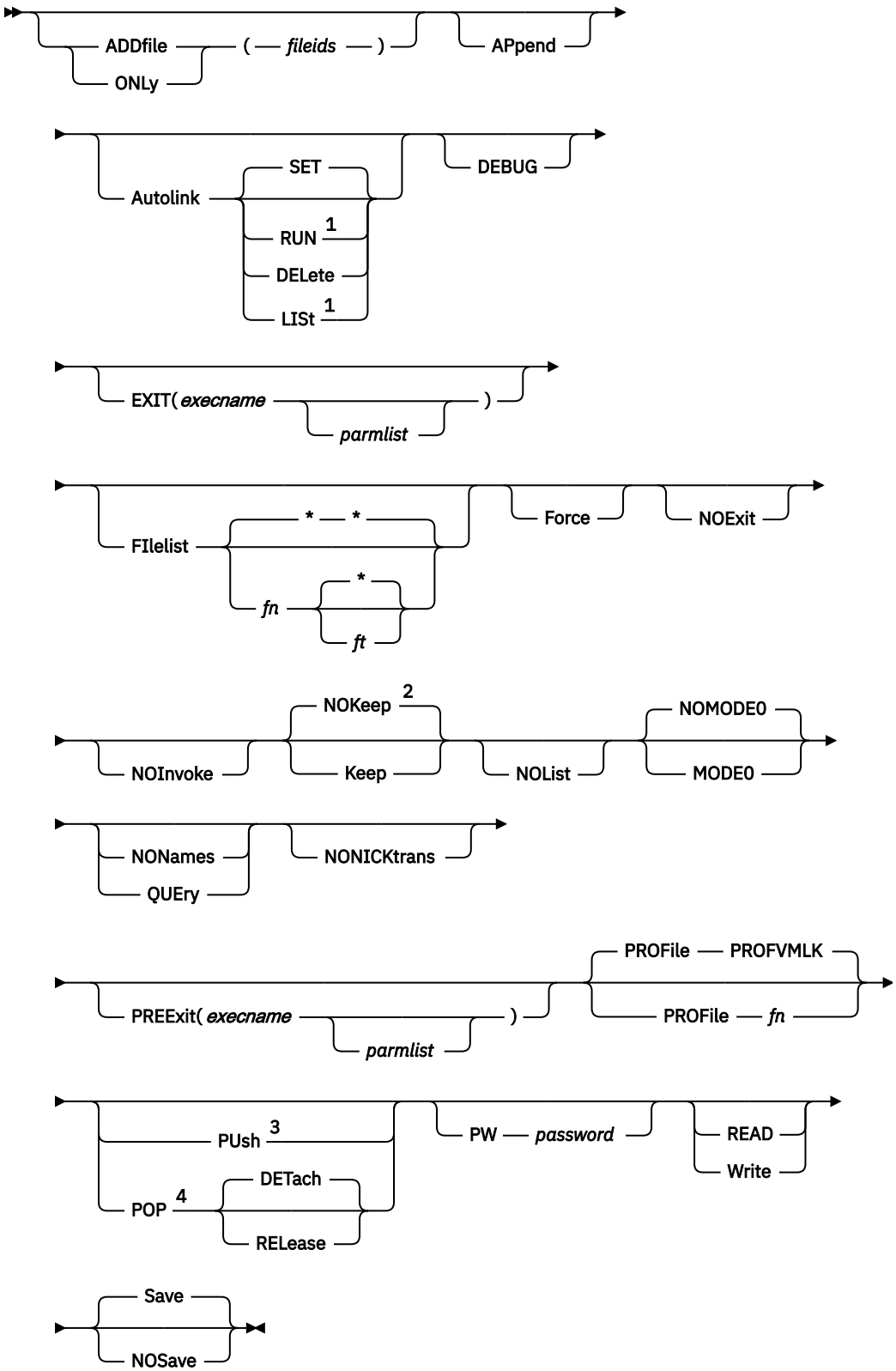
Linking Details



Notes:

[1] If you do not specify any operands, the VMLINK menu is entered unless no NAMES or VMLINK CONTROL files can be accessed, or if none of the nicknames in the NAMES files have a :category tag.
[2] When the POP option is specified, linking details are ignored.
[3] The default options are shown in the Options and Additional Options groups.
[4] When specified, the PUSH option is ignored.
[5] The symbols < and > are part of the VMLINK command Linking Details syntax.

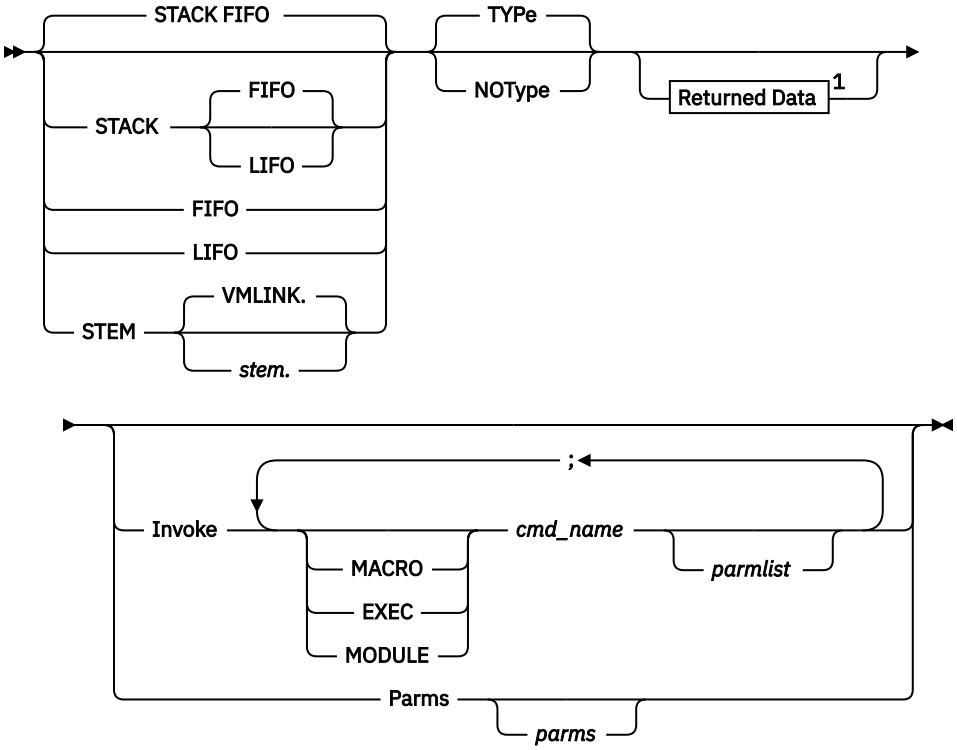
Options



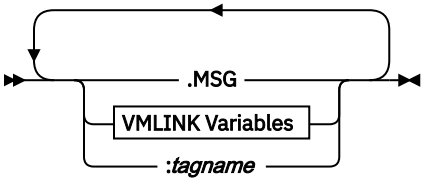
Notes:

[1] When specified, no other options are used.
[2] When NOKEEP is specified, the PUSH option is ignored.
[3] The PUSH option is ignored when DETACH, RELEASE, or NOKEEP is specified.
[4] When the POP option is specified, linking details are ignored.

Additional Options



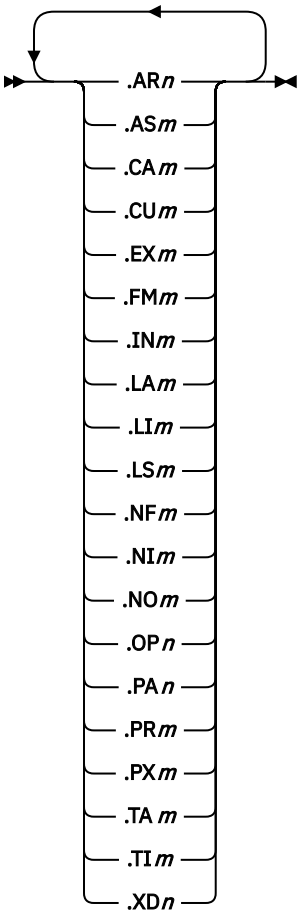
Returned Data



Notes:

[1] Messages, variables, and tag values are returned on the program stack or in REXX stem variables, as indicated by the STACK or STEM keywords.

VMLINK Variables



Authorization

General User

Purpose

Use the VMLINK command to:

- Access and release minidisks
- Link and detach minidisks
- Access and release Shared File System (SFS) directories
- Use nicknames to define minidisks and SFS directories to do all of the tasks defined above

Operands

Disk Operands

nickname
is a nickname defined in a CMS NAMES file by a :NICK tag. For more information on the tags used for VMLINK, see "NAMES" on page 535.

category
is a category name defined in a CMS NAMES file by a :CATEGORY tag. When a category name is specified, the VMLINK panel displays all nicknames in that category. When a disk operand is determined to be a category, any linking details entered are ignored.

.DIR *dirname*
identifies an SFS directory. When an SFS directory is specified this way, use the NONAMES option so VMLINK does not search any NAMES files.

user_ID owner_vdev
identifies the user ID of the owner of a minidisk and the 1-4 character hexadecimal virtual device number as defined in the owner's user directory entry. When a minidisk is specified this way, use the NONAMES option so VMLINK does not search any NAMES files. Otherwise, VMLINK might match the user ID with a nickname. If an asterisk is used for *user_ID*, your user ID is used as the owner.

Linking Details

Note: The symbols < and > must be entered as part of the Linking Details syntax.

vdev
is the 1-4 character hexadecimal virtual device number at which a minidisk is to be linked. To select the next free virtual address in a range, specify the start of a range as *+vdev* or *-vdev* (ascending or descending order). If the disk is already linked and falls in the correct range, the existing virtual device is maintained. For example, if a disk is linked at 200 and +120 is specified as the *vdev*, the 200 link is kept. A *vdev* of 0 is not allowed because one cannot access a minidisk with an address of 0.

An asterisk selects the next free virtual address, as specified in the *VDEV record in the VMLINK CONTROL file, unless the disk is already linked in which case the virtual device the disk is linked at is maintained.

An equal sign tells VMLINK to use the *vdev* range specified on the :PRODUCT tag in the NAMES file if one is found; otherwise, it behaves as an asterisk.

For SFS directories, *vdev* is a place holder.

filemode
is the file mode letter at which the minidisk or SFS directory is to be accessed. You cannot use the file mode S. The file mode letter can be followed by a file mode number. The allowable file mode numbers are 0 to 6. Using a file mode number defines a subset of the files on the specified minidisk or SFS directory. Only the files with the file mode number specified are included in the user file directory and only those files can be read. When a file mode number is used, the minidisk or SFS directory is accessed as an extension of itself and therefore is accessed as R/O. If a file mode number of 0 is used, nothing is accessed unless the MODE0 option is used and the ACCESSM0 command is available and set to ON. To have the first free file mode letter in a range assigned, specify the range as *filemode1-filemode2*. In a range, only a file mode letter is valid for *filemode1*, but *filemode2* can be a file mode letter or a file mode letter and a file mode number. If the disk is already accessed and falls in the correct range, the existing file mode is kept. For example, if a disk is accessed at G and F-K is specified as the file mode range, the G file mode is kept.

An asterisk selects the default range, as specified in the *MODES record in the VMLINK CONTROL file, unless the disk is already accessed, in which case the file mode at which the disk is accessed is kept. The product default in the VMLINK CONTROL file is Z-A, which means the search starts at "Z" and progresses toward "A".

An equals sign tells VMLINK to use the file mode range specified on the :PRODUCT tag in the NAMES file, if one is found; otherwise, it behaves as an asterisk.

If no file mode is available in the range specified, a DMS1281E error results. VMLINK does not access over an already-accessed file mode unless a single file mode is specified, such as P or P/S. If the file mode is specified as P-P or P,P and P is already in use, a DMS1281E error occurs.

/ext
is the *file mode* of the parent minidisk or SFS directory. Files on the minidisk (*vdev*) or directory (*dirname*) 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. An equal sign tells VMLINK to access the minidisk or SFS directory as an extension of itself.

link_mode
specifies whether you get read/only or read/write access to the minidisk or SFS directory. For a minidisk, all modes defined for the CP LINK command are valid. The default is RR.

An asterisk specified for a minidisk link_mode means:

- If no link exists, RR will be used
- If a link already exists, the existing link will be used

For SFS, you can specify FORCERO or FORCERW. The default is:

- FORCERO if you do not own the SFS directory
- FORCERW if you own a file control SFS directory

Note: If you own a directory control SFS directory, FORCERW is the default unless someone else has the SFS directory accessed R/W, in which case FORCERO is the default.

An asterisk specified for an SFS directory link mode follows the default behavior as described above.

If it is unknown what the nickname represents, minidisk or SFS directory, use the READ and WRITE options, instead of *link_mode*, to ensure VMLINK uses the appropriate link modes.

An equal sign tells VMLINK to use the link mode specified on the :PRODUCT tag in the NAMES file if one is found; otherwise, it behaves as an asterisk.

DETach
detaches and releases a minidisk, if linked, or releases an SFS directory, if accessed.

When you detach a minidisk with the READ or WRITE option, VMLINK issues a warning message if you do not have the minidisk linked in the matching mode or if the minidisk remains linked in the other mode. The READ option is the default. When you detach an SFS directory, the SFS directory is released regardless of the mode it is accessed in.

RELease
releases a minidisk or SFS directory if already accessed.

Options

ADDfile(*fileids*)
lists additional NAMES files to be searched before the files in the *FILES control record and overrides the files listed in the *ADDFILE control record in VMLINK CONTROL. More than one file ID can be specified. The *file name* and *file type* can be .NO and will resolve to the user's node ID. The *file type* and mode of the last *file name* in the list can default to `NAMES *`.

ONLy(*fileids*)
specifies which NAMES files should be searched. The ONLY option overrides the *FILES and *ADDFILE control records. More than one file ID can be specified. The *file name* and *file type* can be .NO and will resolve to the user's node ID. The *file type* and mode of the last *file name* in the list can default to `NAMES *`.

APpend
appends the list of nicknames to the list displayed on the VMLINK menu. This option can be used only within the menu.

Autolink
Autolink SET
puts the disk operand in a list of minidisks and/or SFS directories to be accessed when you log on or IPL CMS. The minidisks and/or SFS directories are also linked and accessed when this is issued. If the minidisks and/or SFS directories cannot be linked/accessed, no AUTOLINKs are set. The list is kept in the LASTING GLOBALV file. Options are not saved in the list, so if nicknames are used, they must be in one of the default NAMES files. If this option is entered with no disk operands, it is ignored.

The autolinks are not made when you:

- IPL with the NOSPROF parameter: `ipl cms parm nosprof`

- Enter `access (nodisk` at the VM READ after the IPL is complete

When setting an autolink, you may get the message DMS2064E [RC=1], identifying the GLOBALV line was truncated. If you set many autolinks, you may get this message because the GLOBALV line is too long. See "GLOBALV" on page 366 for information on this size restriction. Each category has its own line, so this problem should only occur when setting many autolinks for nicknames in a given category.

Autolink RUN
links and accesses the minidisks or accesses the SFS directories specified by Autolink SET. If both Autolink RUN and Autolink LIST are specified, only the first one specified is used. Any operands or other options are not used when this option is entered, with the exception of the TYPE and NOTYPE options.

Autolink DELete
deletes a minidisk or SFS directory from the list of things to be accessed when you log on or IPL CMS. Autolink DELETE does not detach or release any minidisk or SFS directories. If this option is entered with no disk operands, it is ignored.

Autolink LISt
displays the autolinks currently set. Here is an example of what is displayed if there is an autolink set for MYNICK:

```
Number Autolink  Setting
1      - MYNICK <= = =>
```

If no autolinks are set, message **DMS739W No autolinks are set** is issued. Any operands or other options are not used when this option is entered.

DEBUG
is provided to help aid in debugging problems. It will help identify from which NAMES file the nickname entry, if any, was obtained. Also, some messages from CP or CMS are displayed. The messages produced for the DEBUG option are for diagnostic purposes only and are not translated.

EXIT(*execname parmlist*)
names an exit exec to be called after each minidisk or SFS directory has been accessed.

Note: For more information, see the PREEXIT option.

FIlelist *fn ft*
causes VMLINK to issue a FILELIST command for files on the first minidisk or SFS directory accessed. If linking or accessing the minidisk or SFS directory fails, FILELIST is not issued. VMLINK returns the accessed minidisks and/or SFS directories to their prior status before it exits; to keep the minidisks and/or SFS directories accessed, use the KEEP option. FILELIST overrides any commands on the :INVOKE tag in the NAMES file. The default is `FILELIST * *`.

When options follow FILELIST, the *fn* and *ft* parameters are required.

Using this option is a shortcut way of saying INVOKE EXEC FILELIST *fn ft .fm1*.

When the POP option is used, VMLINK does not process the FILELIST option.

Force
forces a link at *vdev*. If some other device is attached at *vdev*, it is detached.

NOExit
specifies the :PREEXIT and :EXIT tags in the NAMES file are not used.

NOInvoke
specifies the :INVOKE tags in the NAMES file are not used.

NOKeep
is used with FILELIST or INVOKE to restore minidisks and/or SFS directories to their prior status after VMLINK has finished. This is the default.

NOKEEP has no effect when neither FILELIST nor INVOKE is specified.

If the FORCE option or linking details of either RELEASE or DETACH are specified on the command, it may not be possible to restore the minidisks or directories that have been detached or released by the command.

Keep
is used with FILELIST or INVOKE to keep minidisks and/or SFS directories accessed after VMLINK has finished. It has no effect when FILELIST or INVOKE is not used.

NOList
specifies the :LIST tags in the NAMES file are not used.

NOMODE0
specifies mode 0 links will not be done. This is the default.

MODE0
specifies mode 0 links will be done provided the ACCESSM0 command is available and on. If ACCESSM0 is not available, an error message is displayed.

NONames
suppresses the search of NAMES files. Use NONAMES when the minidisk is identified by *user_ID owner_vdev*, or the SFS directory is identified by *.DIR dirname*.

QUEry
displays all NAMES file entries for each specified nickname regardless of the nodes specified on any :NODE tag. When this option is used the nickname is not processed, therefore no actions are taken on the nickname. All disk operands passed in are considered nicknames when this option is specified.

NONICKtrans
tells VMLINK not to call the VMLNICXT EXEC, which is an exit exec that can be locally customized. The function of this exit exec is to replace a specified nickname with another before the NAMES files are searched. For more information, see "Testing with the VMLNICXT EXEC" in *z/VM: CMS User's Guide*.

If VMLNICXT EXEC exists, it is called unless this option is specified.

PREExit(*execname parmlist*)
names an exit exec to be called after VMLINK has located a free virtual device number and *file mode* letter and before the minidisk or SFS directory is accessed. The parameter list can include all the VMLINK variables except .AS, .LA, and .LS.

Usage Notes for PREExit:

- The exit exec is called separately for each minidisk or SFS directory accessed by the VMLINK command and has access to variable and tag data for the minidisk or SFS directory only.
- A disk identifier is always passed to the exit as the first argument.
- A parameter list passed to an exit cannot include a right parenthesis. The first right parenthesis encountered indicates the end of the exit's parameter list. If a right parenthesis is needed, use the :EXIT or :PREEXIT tags in the NAMES file, rather than the option.
- If a non-zero return code is returned from the exit exec, VMLINK ends. A return code of 12 terminates VMLINK with a return code 0. Otherwise VMLINK terminates with a return code in the form 3*xxx*, where *xxx* is the return code from the exit.
- This option overrides the equivalent tag in the NAMES file.
- It is used in addition to any exit specified in the VMLINK CONTROL file.
- When the POP option, the DETACH operand, or the RELEASE operand is used, VMLINK does not call exit execs.

PROFile *fn*
names the XEDIT profile for the VMLINK panel. The default is PROFVMLK. For more information on the PROFVMLK macro, see Default Key Settings.

PUsh
causes VMLINK to keep a record of its actions when it links and accesses minidisks and SFS directories in read/only mode. This record is kept by stacking the information in LIFO order. A later call to VMLINK for a minidisk or SFS directory with the POP option restores it to its former state.

Note: WRITE disk status is not saved or restored using this option.

POP
POP DETach
causes VMLINK to restore specified minidisks and/or SFS directories to their previous states, based on information saved by linking with the PUSH option. This is done by taking items off the stack in the reverse order of how they were stacked by PUSH; therefore, if an item in the middle of the stack is requested with the POP option, all subsequent items stacked by using the PUSH option will be restored with this POP. If a disk has been detached since the PUSH was done, the disk cannot be restored. If no disk operand is specified with this option, all minidisks and/or SFS directories for which information has been saved are restored.

WRITE disk status is not saved or restored using this option.

When POP is specified, VMLINK ignores linking details and does not call exit execs, run invoke routines, or process the FILELIST option.

POP RELease
causes VMLINK to restore specified minidisks and/or SFS directories to their previous states, based on information saved by linking with the PUSH option, except the minidisks are not detached. This is done by taking items off the stack in the reverse order of how they were stacked by PUSH; therefore, if an item in the middle of the stack is requested with the POP option, all subsequent items stacked by using the PUSH option will be restored with this POP. If a disk has been detached since the PUSH was done, the disk cannot be restored. If no disk operand is specified with this option, all minidisks and/or SFS directories for which information has been saved are restored.

WRITE disk status is not saved or restored using this option.

When POP is specified, VMLINK ignores linking details and does not call exit execs, run invoke routines, or process the FILELIST option.

PW *password*
specifies a password to be used for the CP link command. The password applies to all minidisks linked. For SFS directories, this option is not used.

READ
accesses the minidisk or SFS directory in read/only mode regardless of the NAMES file default. Minidisks are linked with a link mode of RR. SFS directories are accessed with FORCERO. For minidisks, an existing write link to the same minidisk is not disturbed, unless the same virtual device number is provided.

READ overrides any *link_mode* specified in the linking details and limits the effect of DETACH on minidisks to those linked in read mode. If it is unknown what the nickname represents, minidisk or SFS directory, use the READ option, instead of *link_mode*, to ensure VMLINK gets the minidisk or SFS directory in read/only mode.

Write
accesses the minidisk or SFS directory in write mode regardless of the NAMES file default. Minidisks are linked with a link mode of M. SFS directories are accessed with FORCERW. For minidisks, an existing read/only link to the same minidisk is not disturbed, unless the same virtual device number is provided.

WRITE overrides any *link_mode* specified in the linking details and limits the effect of DETACH on minidisks to those linked in write mode. If it is unknown what the nickname represents, minidisk or SFS directory, use the WRITE option, instead of *link_mode*, to ensure VMLINK gets the minidisk or SFS directory in write mode.

Save
causes VMLINK to save the program stack and restore it after VMLINK is done. Any stacked data is not available to the INVOKE routine. This is the default.

NOSave
prevents VMLINK from saving and clearing the program stack, so routines called with the INVOKE option or :INVOKE tag can use the data on the stack.

STACK FIFO
STACK
FIFO
specifies the returned data should be stacked FIFO. This is the default if return data is requested.

STACK LIFO
LIFO
specifies the returned data should be stacked LIFO.

STEM *stem.*
specifies the returned data should be returned to the calling program by setting a stem variable. If a stem name is not provided, *VMLINK.* is used as the default.

Type
displays warning and informational messages on the terminal. This is the default.

NOType
prevents warning and informational messages from being displayed on the terminal.

Invoke *environment cmd_name parmlist*
names routines to be executed after all the minidisks and/or SFS directories have been accessed. If linking or accessing the minidisk or SFS directory fails, the invoke routines are not executed. VMLINK returns the accessed minidisks and/or SFS directories to their prior status before it exits; to keep the minidisks and/or SFS directories accessed, use the KEEP option.

The INVOKE option overrides the :INVOKE tag in the NAMES files.

If used, the INVOKE option must be the last option specified and cannot be used with PARMS.

When the POP option is used, VMLINK does not run invoke routines.

About the parameter list:

- All text following INVOKE is considered part of its parameter list.
- More than one set of *environment cmd_name parmlist* can be used, delimited by semicolons. To pass a literal semicolon as a parameter, specify two adjacent semicolons.
- The environment is optional; the default is to invoke *cmd_name* as though from the CMS command line. Environments that can be specified are:

  MACRO
  to invoke *cmd_name* as an XEDIT macro.

  EXEC
  to invoke *cmd_name* as an exec.

  MODULE
  to invoke *cmd_name* in the command environment (like REXX/VM `address COMMAND`). CP commands must be preceded by "CP", and execs by "EXEC".

If VMLINK receives a non-zero return code from a routine, VMLINK issues message number 2884E with the routine name and a non-zero return code. If multiple routines are invoked, this message is issued for each routine that passes back a non-zero return code to VMLINK. VMLINK terminates with a return code in the form 3*xxx*, where *xxx* is the sum of the non-zero return codes from the invoked routines.

Parms *parms*
specifies parameters on the command invocation to be passed to routines called with the EXIT, PREEXIT, or INVOKE options or the :EXIT, :PREEXIT, or :INVOKE tags. You may use the .PA variable

in a routine's parameter list; the VMLINK command replaces .PA with *parms*. The list of parameters specified with PARMS can include any of the VMLINK variables.

All text following the PARMS option is considered part of the parameter list, so PARMS must be the last option specified.

Returned Data

.MSG
puts VMLINK messages prefixed with *.MSG on the stack if the STACK option was specified or prefixed with .MSG in REXX stem variables if the STEM option was specified. The default is to display messages at the terminal. When messages are stacked or placed in REXX stem variables, the entire message is always provided regardless of the CP EMSG setting. Messages are returned before any other returned data.

:tagname
returns the value assigned to the tag in the names file.

VMLINK Variables

VMLINK Variables can be used as part of the parameter lists for PARMS, invoke routines, pre exits and exits.

.AR*n*
returns the operands specified on the VMLINK command. The arguments include everything preceding the first left parenthesis if there is one.

.AS*m*
returns the access status, either R/O (read/only) or R/W (read/write), of the minidisk or SFS directory that was accessed. The .AS option is ignored in the parameter list of a PREEXIT EXEC because the minidisk or SFS directory is not yet accessed.

.CA*m*
returns the :CATEGORY tag value.

.CU*m*
returns the virtual device number used to link a minidisk, or a string containing "DIR" followed by the fully qualified name of the SFS directory that was accessed. For example, `DIR SYSSERV:JQUSER.TASKS`.

.EX*m*
returns the :EXIT tag value.

.FM*m*
returns the single character *file mode* letter used to access the minidisk or SFS directory, omitting the *file mode* extension used.

.IN*m*
returns the :INVOKE tag value.

.LA*m*
returns the label of a minidisk or a hyphen for an SFS directory. The .LA option is ignored in the parameter list of a PREEXIT EXEC, because the minidisk or SFS directory is not yet accessed.

.LI*m*
returns the :LIST tag value.

.LS*m*
returns the link status, either R/O (read/only) or R/W (read/write), of the minidisk that was linked, or DIR for an SFS directory. The .LS option is ignored in the parameter list of a PREEXIT EXEC because the minidisk or SFS directory is not yet linked.

.NF*m*
returns the file ID of the NAMES file in which VMLINK found the nickname.

.NI*m*
returns the :NICK tag value.

.NO*m*
returns the :NODE tag value.

.OP*n*
returns the options passed with VMLINK. The options include everything following the first left parenthesis.

.PA*n*
returns the text following the PARMS option keyword.

.PR*m*
returns the :PRODUCT tag value, or if VMLINK was invoked with the explicit minidisk or SFS directory name, it will return *user ID vdev linking details* or *.DIR dirname linking details*

.PX*m*
returns the :PREEXIT tag value.

.TA*m*
returns the tag-value pairs for all *:tagname* options specified on the command invocation of VMLINK.

.TI*m*
returns the :TITLE tag value.

.XD*n*
returns all the data specified on the RETURN statement of any exit execs.

n
identifies a particular blank-delimited token in the requested string. For example, .OP3 refers to the third token in the options string. If *n* is not specified, the complete string is returned.

m
identifies the minidisk or SFS directory for which the information is requested. For example, .FM3 is the *file mode* letter of the third minidisk or SFS directory accessed by the VMLINK command. If *m* is not specified, 1 is assumed.

Usage Notes

1. For more information on how to customize this command, see Appendix A, “Customizing Profiles for CMS Productivity Aids,” on page 1419.
2. VMLINK determines how to process each disk operand in turn, by trying to establish one of these conditions as true:
   a. The NONAMES option is specified. If so, the operands are assumed to be only minidisk IDs and SFS directory names. No NAMES file is searched for nicknames.
   b. The operand is a nickname that is valid on the node. If so, VMLINK processes it using a NAMES file.
   c. The operand matches the value of a :CATEGORY tag that is valid for the node. If so, a menu of all the valid nicknames with a matching category is displayed. When an operand is determined to be a category, any linking details entered are ignored.
   d. The first word of the disk operand is ".DIR". If so, VMLINK processes the operand as part of an SFS directory specification.
   e. Otherwise, VMLINK tries to process the operand as part of a minidisk ID in the form *user_ID vdev*.
3. VMLINK searches CMS NAMES files for nicknames. The search order and names of the NAMES files are defined in the VMLINK CONTROL file or specified on the command. For more information on the VMLINK NAMES file, see *z/VM: CMS User's Guide*.
4. The VMLINK control file, VMLINK CONTROL, sets local defaults for some VMLINK values. The defaults are overridden by values in a nickname entry, which in turn are overridden by operands specified with the command. For more information on the VMLINK CONTROL file, see *z/VM: CMS User's Guide*.
5. A disk identifier is always passed as the first argument to an exit exec. The form of the disk identifier is determined by the way the minidisk or SFS directory is specified in the command.

Table 52. Disk Identifiers

Specification	ID format	Disk ID
proda191	**nickname*	*PRODA191
productc 193	**USERIDvdev*	*PRODUCTC193
.dir maint.pets	*.DIR*dirname*	*.DIRMAINT.PETS
.dir .pets	*.DIR*dirname*	*.DIR.PETS

6. If you have called VMLINK with the NONAMES option or the NONICKtrans option, no VMLNICXT EXEC is called.

Examples

1. **VMLINK Autolink**

   VMLINK has the ability to link and access minidisks or access SFS directories automatically when a user re-IPLs CMS. The standard SYSPROF EXEC executes 'EXEC VMLINK (AUTOLINK RUN'. AUTOLINK RUN tells VMLINK to check the LASTING GLOBALV for any definitions set with the VMLINK (AUTOLINK SET command. Upon finding a definition, VMLINK performs the operation during the execution of the SYSPROF EXEC.

   In Figure 52 on page 1155, you will see a console listing of setting an autolink, IPLing CMS, and then deleting the autolink:

```
vmlink .dir vmsysu:maint.samples <= J-H> ( autolink set nonames
VMSYSU:MAINT.SAMPLES accessed as filemode J
Autolink status updated for VMSYSU:MAINT.SAMPLES
Ready; T=0.08/0.08 13:43:31
q search
MAINT   191   A    R/W
-       DIR   J    R/W   VMSYSU:MAINT.SAMPLES
MNT190  190   S    R/O
Y-DISK  19E   Y/S  R/O
Ready; T=0.01/0.01 13:43:36
i cms
z/VM ....

VMSYSU:MAINT.SAMPLES accessed as filemode J
Ready; T=0.12/0.14 13:43:40
vmlink .dir vmsysu:maint.samples ( autolink del
Autolink removed for .DIR VMSYSU:MAINT.SAMPLES
Ready; T=0.04/0.04 13:43:58
```

   Figure 52. VMLINK Autolink Example

   It is important to notice in Figure 52 on page 1155 that VMLINK accessed the samples when setting the autolink. Also, the "=" sign was used as a place holder so access modes J-H could be specified with an SFS directory.

2. a. **PUSH and POP Option Examples**

   Figure 53 on page 1155 and Figure 54 on page 1156 show the status of minidisks and/or SFS directories accessed and linked before VMLINK is run:

```
Q DISK

LABEL VDEV M   STAT  CYL TYPE BLKSIZE FILES BLK USED-(%) BLK LEFT BLK TOT
XX191 191  A    R/W   10 3380 4096    256        913-61       587    1500
CMS21 190  S    R/O   96 3390 4096   1037      15915-53     14085   30000
YDISK 19E  Y/S  R/O  300 3380 4096   1550      30312-67     14688   45000
```

   Figure 53. Example of minidisk Links Before VMLINK

```
Q DASD

DASD 0190 3390 RSE702 R/O     100 CYL ON DASD   1755 SUBCHANNEL = 000F
DASD 0191 3380 USE735 R/W      10 CYL ON DASD   183C SUBCHANNEL = 000A
DASD 019E 3380 SYE7MC R/O     300 CYL ON DASD   183A SUBCHANNEL = 0011
```

Figure 54. Example of DASD Links Before VMLINK

```
VMLINK DITTO <* A> (PUSH
```

You would link DITTO as VDEV (virtual device) 120 *file mode* A.

Figure 55 on page 1156 and Figure 56 on page 1156 show the status of minidisks accessed and linked and/or SFS directories accessed after VMLINK with PUSH option:

```
Q DISK

LABEL VDEV M   STAT  CYL TYPE BLKSIZE FILES BLK USED-(%) BLK LEFT BLK TOT
DITTO 120  A    R/O   12 3380 4096     40       1065-59       735    1800
CMS21 190  S    R/O   96 3390 4096   1037      15915-53     14085   30000
YDISK 19E  Y/S  R/O  300 3380 4096   1550      30312-67     14688   45000
```

Figure 55. Example of minidisk Links After VMLINK (PUSH

```
Q DASD

DASD 0120 3380 CMSY74 R/O      12 CYL ON DASD   1830 SUBCHANNEL = 001A
DASD 0190 3390 RSE702 R/O     100 CYL ON DASD   1755 SUBCHANNEL = 000F
DASD 0191 3380 USE735 R/W      10 CYL ON DASD   183C SUBCHANNEL = 000A
DASD 019E 3380 SYE7MC R/O     300 CYL ON DASD   183A SUBCHANNEL = 0011
```

Figure 56. Example of DASD Links After VMLINK (PUSH

If you entered:

```
VMLINK DITTO (POP RELEASE
```

The status of the minidisks accessed and linked would be as shown in Figure 57 on page 1156 and Figure 58 on page 1156:

```
Q DISK

LABEL VDEV M   STAT  CYL TYPE BLKSIZE FILES BLK USED-(%) BLK LEFT BLK TOT
XX191 191  A    R/W   10 3380 4096    256        913-61       587    1500
CMS21 190  S    R/O   96 3390 4096   1037      15915-53     14085   30000
YDISK 19E  Y/S  R/O  300 3380 4096   1550      30312-67     14688   45000
```

Figure 57. Example of minidisk Links After VMLINK (POP RELEASE

```
Q DASD

DASD 0120 3380 CMSY74 R/O      12 CYL ON DASD   1830 SUBCHANNEL = 001A
DASD 0190 3390 RSE702 R/O     100 CYL ON DASD   1755 SUBCHANNEL = 000F
DASD 0191 3380 USE735 R/W      10 CYL ON DASD   183C SUBCHANNEL = 000A
DASD 019E 3380 SYE7MC R/O     300 CYL ON DASD   183A SUBCHANNEL = 0011
```

Figure 58. Example of DASD Links After VMLINK (POP RELEASE

Notice the 191 minidisk is the A-disk again as it was before VMLINK was entered, but the 120 minidisk is still linked because POP RELEASE was used. This will mean better performance for subsequent VMLINKs for this minidisk because the minidisk will only have to be accessed and not linked again.

b. **Multiple PUSH and POP Option Example**

Figure 59 on page 1157 and Figure 60 on page 1157 show the status of minidisks and/or SFS directories accessed and linked before VMLINK is run:

```
Q DISK

LABEL VDEV M   STAT  CYL TYPE BLKSIZE FILES BLK USED-(%) BLK LEFT BLK TOT
XX191 191  A    R/W   10 3380 4096    256        913-61       587    1500
CMS21 190  S    R/O   96 3390 4096   1037      15915-53     14085   30000
YDISK 19E  Y/S  R/O  300 3380 4096   1550      30312-67     14688   45000
```

Figure 59. Example of minidisk Links Before VMLINK

```
Q DASD

DASD 0190 3390 RSE702 R/O     100 CYL ON DASD   1755 SUBCHANNEL = 000F
DASD 0191 3380 USE735 R/W      10 CYL ON DASD   183C SUBCHANNEL = 000A
DASD 019E 3380 SYE7MC R/O     300 CYL ON DASD   183A SUBCHANNEL = 0011
```

Figure 60. Example of DASD Links Before VMLINK

```
VMLINK DITTO1 <* B> (PUSH
VMLINK DITTO2 <* A> (PUSH
VMLINK DITTO3 <* C> (PUSH
```

DITTO1 is linked as 120 and accessed at *file mode* B.
DITTO2 is linked as 121 and accessed at *file mode* A.
DITTO3 is linked as 122 and accessed at *file mode* C.

Figure 61 on page 1157 and Figure 62 on page 1157 show the status of minidisks accessed and linked and/or SFS directories accessed after VMLINK with PUSH option:

```
Q DISK

LABEL VDEV M   STAT  CYL TYPE BLKSIZE FILES BLK USED-(%) BLK LEFT BLK TOT
DITTO2 121 A    R/O   12 3380 4096     40       1065-59       735    1800
DITTO1 120 B    R/W   10 3380 4096    250        900-59       600    1500
DITTO3 122 C    R/W   10 3380 4096    256        913-61       587    1500
CMS21  190 S    R/O   96 3390 4096   1037      15915-53     14085   30000
YDISK  19E Y/S  R/O  300 3380 4096   1550      30312-67     14688   45000
```

Figure 61. Example of minidisk Links After VMLINK (PUSH

```
Q DASD

DASD 0120 3380 USE800 R/W      10 CYL ON DASD   2200 SUBCHANNEL = 000A
DASD 0121 3380 CMSY74 R/O      12 CYL ON DASD   1830 SUBCHANNEL = 001A
DASD 0122 3380 USE800 R/W      10 CYL ON DASD   2400 SUBCHANNEL = 000A
DASD 0190 3390 RSE702 R/O     100 CYL ON DASD   1755 SUBCHANNEL = 000F
DASD 0191 3380 USE735 R/W      10 CYL ON DASD   183C SUBCHANNEL = 000A
DASD 019E 3380 SYE7MC R/O     300 CYL ON DASD   183A SUBCHANNEL = 0011
```

Figure 62. Example of DASD Links After VMLINK (PUSH

If you entered:

```
VMLINK DITTO2 (POP
```

The status of the minidisks accessed and linked would be as shown in Figure 63 on page 1158 and Figure 64 on page 1158:

```
Q DISK

LABEL VDEV M   STAT  CYL TYPE BLKSIZE FILES BLK USED-(%) BLK LEFT BLK TOT
XX191 191  A    R/W   10 3380 4096    256        913-61       587    1500
DITTO1 120 B    R/W   10 3380 4096    250        900-59       600    1500
CMS21 190  S    R/O   96 3390 4096   1037      15915-53     14085   30000
YDISK 19E  Y/S  R/O  300 3380 4096   1550      30312-67     14688   45000
```

Figure 63. Example of minidisk Links After VMLINK (POP

```
Q DASD

DASD 0120 3380 USE800 R/W      10 CYL ON DASD   2200 SUBCHANNEL = 000A
DASD 0190 3390 RSE702 R/O     100 CYL ON DASD   1755 SUBCHANNEL = 000F
DASD 0191 3380 USE735 R/W      10 CYL ON DASD   183C SUBCHANNEL = 000A
DASD 0191 3380 USE735 R/W      10 CYL ON DASD   183C SUBCHANNEL = 000A
DASD 019E 3380 SYE7MC R/O     300 CYL ON DASD   183A SUBCHANNEL = 0011
```

Figure 64. Example of DASD Links After VMLINK (POP

Notice that because POP was issued against DITTO2, which was the second name PUSHed, both DITTO2 and DITTO3 were restored to their previous state. Also notice the 191 minidisk is the A-disk again, as it was before VMLINK DITTO2 was entered.

3. **Simple Calls to VMLINK**

   The simplest call to VMLINK is to access one minidisk using the *user_ID vdev* disk operands.

   ```
   VMLINK PRODUCTC 193 (NONAMES
   ```

   VMLINK will not search any NAMES files but will link and access the minidisk, using the next free virtual device number and *file mode* letter.

   The simplest call to VMLINK via a nickname would look as follows:

   ```
   VMLINK GDDM
   ```

   VMLINK searches the NAMES files for `:nick.gddm` and accesses the minidisks and/or SFS directories, using the next free virtual device numbers and *file mode* letters.

   To detach the minidisks and/or SFS directories with the nickname GDDM®, enter

   ```
   VMLINK GDDM <DETACH>
   ```

   VMLINK searches the NAMES files for `:nick.gddm` and releases and detaches the GDDM minidisks and releases the GDDM SFS directories.

   If you want just to release the minidisks and/or SFS directories, use RELEASE instead of DETACH:

   ```
   VMLINK GDDM <RELEASE>
   ```

4. **Complex VMLINK Call**

   You can specify different linking details for different nicknames. The linking details apply only to the nickname immediately preceding them.

   ```
   VMLINK GDDM SPF <DETACH> APL2 <* */A>
   ```

   VMLINK searches the NAMES file for the tags `:nick.GDDM`, `:nick.SPF`, and `:nick.APL2`, and then:

   - Links and accesses the GDDM minidisks and accesses the GDDM SFS directories using the default virtual device number, access mode, and link mode.
   - Releases and detaches the SPF minidisks or releases the SPF SFS directories.
   - Links the APL2® minidisks starting at the default virtual device number and accesses the minidisks and/or SFS directories starting at the default *file mode* letter, each as an extension of *file mode* A.

5. **VMLINK Minidisks by Name**

When a minidisk is specified by name, rather than nickname, use the NONAMES option so VMLINK does not search any NAMES files:

```
VMLINK MAINT 195 <RELEASE> (NONAMES
```

releases the MAINT 195 minidisk.

6. **Option Examples**

   These examples apply to calls from the command line and from execs.

   ADDFILE

   ```
   VMLINK MYDISK (ADDFILE(MYNAMES NAMES A TOOLS)
   ```

   searches MYNAMES NAMES A, TOOLS NAMES *, and then the rest of the NAMES files as specified in the control file.

   ONLY

   ```
   VMLINK MYDISK ( ONLY(MYNAMES NAMES A TOOLS)
   ```

   searches only MYNAMES NAMES A and TOOLS NAMES *.

   INVOKE

   ```
   VMLINK GDDM  (KEEP INVOKE GLOBAL TXTLIB ADMGLIB ADMRLIB ; ADMCHART
   ```

   invokes two commands, `GLOBAL TXTLIB ADMGLIB ADMRLIB` and `ADMCHART`, after linking and accessing minidisks and/or accessing SFS directories associated with GDDM.

   KEEP

   The KEEP option causes VMLINK to leave the GDDM minidisk linked and accessed and/or the GDDM SFS directories accessed.

   ```
   VMLINK GDDM  (INVOKE COPYFILE ADMDEFS PROFILE .FM1 = = A ( REP
   ```

   accesses the GDDM minidisks and/or SFS directories and copies ADMDEFS PROFILE from the first GDDM minidisk or directory (*file mode* .FM) to *file mode* A.

   PARMS

   Suppose there is this NAMES entry:

   ```
   :nick.MYDISK
   :product.MAINT 200 .<* Q-Z>
   :invoke.MYEXEC .FM  .PA
   ```

   If the command:

   ```
   VMLINK MYDISK (NOTYPE PARMS DOG CAT HAMSTER
   ```

   links MAINT 200 at Q, then the INVOKE option executes

   ```
   MYEXEC Q  DOG CAT HAMSTER
   ```

7. **VMLINK XEDIT Panel**

   XEDIT Environment

   When you invoke the VMLINK command with no operands, VMLINK displays a list of nicknames.

   The full power of XEDIT is available to you while you issue commands against the list of nicknames.

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

Symbols Used in the Panel Fields

On the panel, the linking details specified in the NAMES file entry appear in the `Vdev`, `Fm`, `Ext`, and `Lm` fields. An equals sign indicates nothing is specified and the default value is being used.

Field
Contents

Vdev
A virtual device number or the start of a range of virtual device numbers, as specified in the nickname entry; an equals sign (=) indicates the default range is used; or "DIR", indicates an SFS directory.

Fm
A file mode letter as specified in the nickname entry; a percent sign (%) indicates a range; or an equals sign indicates the default range is used.

Ext
A file mode letter, as specified in the nickname entry, or a blank indicates nothing was specified.

Lm
A linking mode, as specified in the nickname entry, or a blank indicates nothing was specified.

Entering CMS commands from VMLINK

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

Saving a List of Nicknames

You can save a list of nicknames created by the VMLINK command simply by issuing FILE or SAVE from the command line. The default *file name* is your user ID and the default *file type* is VMLINK. The list replaces any "*user_ID* VMLINK" on your file mode A

You can also save a list of nicknames under a different file ID.

Issuing Commands From the List

On a full screen display, you can issue commands directly from the line on which a nickname is displayed. You do this by moving the cursor to the line that describes the nickname, and typing the command in the space provided to the left of the nickname.

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 are finished typing the command, erase the rest of the line by pressing the ERASE EOF key, or space over the rest of the line. Then press Enter. You may also use the DELETE key to erase the rest of the line, but do not use it to erase only part of the rest of the line. For more information, see “EXECUTE” on page 1393.

When you press Enter, all commands typed on one screen are executed, and the screen is restored to its previous state except the list is updated to reflect the current status of the nicknames.

You may want to enter commands from the VMLINK command line before executing commands 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.

You can use the special command EXECUTE from the VMLINK screen. The EXECUTE command allows you to issue commands that use the nicknames displayed by VMLINK.

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 *nickname*.

These symbols can be used:

/
specifies the *nickname*, <*vdev, file mode, and link mode*>.

Note: The <>'s shown above will be part of what is substituted in the command to be issued.

/n
specifies the *nickname*.

/v
specifies the *vdev*.

/m
specifies the *file mode*.

/l
specifies the *link mode*.

/o
specifies to execute the line as is, and omit appending anything.

Any combinations of symbols can be used. For example:

/n /v
specifies the *nickname* followed by *vdev*.

/nv
specifies the *nickname* followed by *vdev*.

/nvm
specifies the *nickname, vdev, and file mode*.

/vn
specifies the *vdev* followed by the *nickname*.

/nnm
specifies the *nickname* followed by *nickname* and *file mode*.

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.

File Sharing Considerations

While you have your VMLINK menu displayed on the screen, other users may modify the NAMES files used by VMLINK. As these changes are saved or filed, some of the NAMES file tags on your screen may be out of date. When you issue a command against such an entry, you may receive a message:

```
DMS654E  Invalid symbol symbol; {/0 must be specified alone|
           invalid character char following / symbol} [RC=24]
```

If you do receive this message, clear the rest of the line following your command and press Enter.

Note: Always clearing the line following your command will prevent this message from ever occurring.

Special Symbols Used Alone

The following special symbols can be typed alone on the lines of the VMLINK display. They have the following meanings:

=
means execute the previous command for this nickname. Commands are executed starting at the top of the screen. For example, suppose you enter the VMLINK command on the top line. You can then type an equal sign on any other lines. Those nicknames preceded by equal signs are linked when the EXECUTE command is entered from the command line or by 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 VMLINK screen, the current line is the first nickname on the screen.)

L
means link and access the minidisks and/or access the SFS directories for this nickname entry. It will not process an *:invoke* tag. If no minidisks are eligible for linking, or no SFS directories are eligible for accessing, a message will be produced.

I
means link and access the minidisks and/or access the SFS directories for this nickname entry. It will process an *:invoke* tag if one is present. If there is no *:invoke* tag, the minidisks will will remain linked and/or the SFS directories will remain accessed. If no minidisks are eligible for linking, and/or no SFS directories are eligible for accessing, a message will be produced.

D
means release and detach the minidisks and/or release the SFS directories for this nickname entry.

A
means link and access the minidisks and/or access the SFS directories for this nickname entry and establish an autolink entry in the LASTING GLOBALV file.

R
means remove an autolink request for this nickname entry.

Default Key Settings

When you enter the VMLINK command with no operands, VMLINK executes the PROFVMLK XEDIT macro, unless you specify a different macro as an option in the VMLINK command. Table 53 on page 1162 and Table 54 on page 1163 show the values to which the keys are set by PROFVMLK XEDIT.

VMLINK Panel Interface Example

Figure 65 on page 1162 is an example of a VMLINK panel when system support has many products installed. The panel was obtained by entering the VMLINK command with no options specified.

```
 MYUSERID VMLINK   A0  V 260  Trunc=260 Size=188 Line=56 Col=1 Alt=0
Cmd   Nickname Vdev  Fm Ext Lm  Category  Description
      CDSDSK   +200  * /        ALL       CDS Shadow disk
      CFSEARCH +200  P /        ALL       Contextual File Search
      CLIB200   =    = /        OFFICE    ECFORMS Processing Code Disk
      CLIB300   =    = /        OFFICE    ECFORMS Forms Design Code Disk
      CLIB400   =    = /        OFFICE    ECFORMS Forms Admin. Code Disk
      COBOL2    =    = /        ALL       Cobol II
      COLIS    +300  * /        ALL       Colis, PHONE, CALLUP disk
      CORPDIR   =    = /        OFFICE    Corporate Directives
      CSP191   +291  E / A      ALL       A-disk
      CSP193   +293  B /        ALL       B-disk
      CSP195   +295  M /        ALL       M-disk
      CSP502   +202  C /        ALL       VSAM C-disk
      CSP503   +203  D /        ALL       VSAM D-disk
      DEVGUIDE +203  * /        ALL       I/S Development Guide
      DEVSQL   +222  Q /        ALL       SQL for IMF development
      DFSORT    =    = /        ALL       DFSORT
      DMS      +200  * /        ALL       DMS
1= Help     2= Refresh  3= Quit      4= Sort(name)  5= Link      6= Alt PF
7= Backward 8= Forward  9= Category 10= Detach     11= Filelist 12= Cursor

====>
                                                      X E D I T  1 File
```

Figure 65. VMLINK Panel Interface

The VMLINK panel menu is self-explanatory. Some functions are available on the PF Keys, as shown in Table 53 on page 1162:

Table 53. Set 1 PF Keys Assigned by PROFVMLK XEDIT

Key	Setting	Action
Enter	Execute	Execute commands typed on file lines or on the command line.
PF1	Help	Display VMLINK command description.

Table 53. Set 1 PF Keys Assigned by PROFVMLK XEDIT (continued)

Key	Setting	Action
PF2	Refresh	Update the list to indicate new or changed nickname entries, using the same parameters as those specified when VMLINK was invoked.
PF3	Quit	Exit from VMLINK.
PF4	Sort (name)	Sort by nickname.
PF5	Link	Link and access the minidisks using the vdev and mode specified on the menu, and/or access the SFS directories using the mode specified on the menu; if none specified, the defaults will be used.
PF6	Alt PF	Toggle to PF key set 2.
PF7	Backward	Scroll back one screen.
PF8	Forward	Scroll forward one screen.
PF9	Category	Displays an XEDIT panel of category names.
PF10	Detach	Detach the minidisks from your virtual machine and/or release the SFS directories defined by the selected nickname.
PF11	Filelist	Links and accesses the minidisks and/or accesses the SFS directories defined by the selected nickname and issues the Filelist command for all the files on the first minidisk or SFS directory. The minidisks and/or SFS directories will be restored when exiting Filelist.
PF12	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: PF keys 13 to 24 are assigned the same values as PF keys 1 to 12.

PF6 toggles to an alternate set of PF keys (PF key set 2). Figure 66 on page 1163 shows the alternate set of PF keys.

```
      DEVSQL   +222  Q /        ALL       SQL for IMF development
      DFSORT    =    = /        ALL       DFSORT
      DMS      +200  * /        ALL       DMS
1= Help       2= Refresh  3= Quit      4= Sort(desc)  5= Sort(cat)   6= Alt PF
7= Backward   8= Forward  9= Autolink 10= Remove/A   11= Filelist   12= Cursor
```

Figure 66. VMLINK Panel Interface - PF Key Set 2

These alternate PF keys are defined in Table 54 on page 1163:

Table 54. Set 2 PF Keys Assigned by PROFVMLK XEDIT

Key	Setting	Action
Enter	Execute	Execute commands typed on file lines or on the command line.
PF1	Help	Display VMLINK command description.
PF2	Refresh	Update the list to indicate new or changed nickname entries, using the same parameters as those specified when VMLINK was invoked.
PF3	Quit	Exit from VMLINK.

Table 54. Set 2 PF Keys Assigned by PROFVMLK XEDIT (continued)

Key	Setting	Action
PF4	Sort (desc)	Sort by description.
PF5	Sort (cat)	Sort by category.
PF6	Alt PF	Toggle to PF key set 1.
PF7	Backward	Scroll back one screen.
PF8	Forward	Scroll forward one screen.
PF9	Autolink	Establish an autolink for the minidisks and/or SFS directories defined by the selected nickname. The autolink is saved in the LASTING GLOBALV file.
PF10	Remove/A	Remove an autolink for the minidisks and/or SFS directories defined by the selected nickname.
PF11	Filelist	Links and accesses the minidisks and/or accesses the SFS directories defined by the selected nickname and issues the FILELIST command for all the files on the first minidisk or SFS directories. The minidisks and/or SFS directories will be restored when exiting FILELIST.
PF12	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: PF keys 13 to 24 are assigned the same values as PF keys 1 to 12.

In addition to setting the above PF keys, the PROFVMLK XEDIT macro sets synonyms you can use to sort your VMLINK nicknames. The synonyms are:

SCAT
Sorts the list alphabetically by category name.

SDESC
Sorts the list alphabetically by description.

SLMOD
Sorts the list in descending order by link mode.

SMODE
Sorts the list in descending order by access mode.

SNAME
Sorts the list alphabetically by nickname.

SVDEV
Sorts the list in descending order by hexadecimal virtual device number.

Default Key Settings for Category Panel

Pressing PF9 from the VMLINK panel takes you to the Category panel. The function keys are set by the PROFCLST XEDIT macro, as shown in Table 55 on page 1165.

The PF9 key (as shown in Figure 65 on page 1162), displays a panel interface called the **category** panel. Figure 67 on page 1165 has been configured for only two categories for the products: ALL and OFFICE. Additional category names can also be used, such as GRAPHICS, NEWS, TOOLS, and FORUMS.

```
 MYUSERID CATLIST  A0  V 80  Trunc=80 Size=1 Line=1 Col=1 Alt=0
Category Name
ALL
OFFICE


1= Help       2= Refresh  3= Quit   4=            5=             6=
7= Backward   8= Forward  9=       10=           11= VMLINK     12= Cursor
```

Figure 67. Category VMLINK Panel Interface

The category display has a new set of PF keys that are defined in Table 55 on page 1165:

Table 55. Default Key Settings in Category Panel

Key	Setting	Action
PF1	Help	Display VMLINK command description.
PF2	Refresh	Update the list to indicate new or changed nickname entries, using the same parameters as those specified when the category panel was entered.
PF3	Quit	Exit from the category panel.
PF4		Not assigned.
PF5		Not assigned.
PF6		Not assigned.
PF7	Backward	Scroll back one screen.
PF8	Forward	Scroll forward one screen.
PF9		Not assigned.
PF10		Not assigned.
PF11	VMLINK	Issues a VMLINK for the selected category name.
PF12	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: PF keys 13 to 24 are assigned the same values as PF keys 1 to 12.

Messages and Return Codes

- DMS002E File *fn ft fm* not found [RC=28]
- DMS029E Invalid parameter *parameter* in the option *option* field [RC=24]
- DMS066E *option1* and *option2* are conflicting options [RC=24]
- DMS069E Directory *dirname* not accessed [RC=36]
- DMS072E Error in EXEC file *fn* line *n* - *message* [RC=99]
- DMS104S Error *nn* reading file *fn ft fm* from disk or directory [RC=*rc*]
- DMS389E Invalid *operandtype*: *operand* [RC=24]
- DMS394E Invalid option: *option* [RC=24]
- DMS637E Missing value for the *option* option [RC=24]
- DMS651E APPEND must be issued from VMLINK [RC=40]
- DMS739W No autolinks are set

- DMS926E Command is only valid on a display terminal [RC=88]
- DMS1227E No {file mode | virtual device address} is available to {access | link} {*nickname* minidisk | directory} [RC=40]
- DMS1233E Invalid use of *option* option [RC=24]
- DMS2059E USERID VDEV must be specified with the NONAMES option [RC=24]
- DMS2059E .DIR DIRNAME must be specified with the NONAMES option [RC=24]
- DMS2060I {*nickname*|*user ID vdev*|*directory_name*} {linked|accessed} [*vdev*|*link_mode*] as [*vdev*|*link_mode*] filemode *fm* [RC=0]
- DMS2061I *description* [detached | released] [RC=0]
- DMS2062I NAMEFIND search results for file: *fn ft fm* for nickname: *nickname* [RC=0]
- DMS2064I Autolink [status updated|removed] for *nickname* [RC=0]
- DMS2064W Autolink not found for *nickname* [RC=4]
- DMS2064E Autolink update failed RC=*rc* for *nickname* [RC=*rc*]
- DMS2065E Minidisk virtual address *vdev* already defined [RC=40]
- DMS2066E [Virtual address | Link mode *lm*] is not valid [RC=24]
- DMS2067E Unknown [disk nickname (*nickname*) | USER ID (*user ID*) | category (*category*)] [RC=32]
- DMS2068E Disk nickname (*nickname*) not valid on this node id [RC=32]
- DMS2069E No NAMES file(s) found to search for nickname (*nickname*) [RC=28]
- DMS2070E POP data not available [RC=40]
- DMS2071E The MODE0 option cannot be used because the ACCESSM0 command is not available [RC=40]
- DMS2072E No nickname was specified with the QUERY option [RC=24]
- DMS2073W Warning: Duplicate autolink filemode: *fm*
- DMS2074W Warning: Disk *nickname* will be released
- DMS2075W Disk is still accessed {R/W | R/O}. Use the {READ | WRITE} option to {detach | release} [RC=4]
- DMS2076W There was no {R/O | R/W} {disk | directory} to {detach | release} [RC=4]
- DMS2077E FILELIST not done. No disk was accessed or first disk was released [RC=32]
- DMS2884E Unexpected return code *rc* {on command *command* | from exit *exit*} [RC=*rc*]
- DMS2923E Missing right parenthesis [RC=24]

In some cases, the VMLINK return code is determined by a nonzero return code from a command or routine:

RC
Meaning

1*xxx*
CP LINK returned RC = *xxx*.

-1*xxx*
CP LINK returned RC = -*xxx*.

2*xxx*
CMS ACCESS or RELEASE returned RC = *xxx*.

-2*xxx*
CMS ACCESS or RELEASE returned RC = -*xxx*.

3*xxx*
A routine on an INVOKE, EXIT, or PREEXIT option or an :INVOKE, :EXIT, or :PREEXIT tag returned RC = *xxx*.

-3xxx
A routine on an INVOKE, EXIT, or PREEXIT option or on an :INVOKE, :EXIT, or :PREEXIT tag returned RC = *-xxx*.

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411
Errors in the Shared File System	“File Pool Server Messages” on page 1414

VMSIZE

Authorization

General User

Purpose

Use the VMSIZE command to determine the size of a virtual machine's virtual storage in kilobytes. This value is returned as the CMS return code, as produced by DIAGNOSE code X'60'. For example, if issued in a 1 megabyte virtual machine, the return code from VMSIZE is 1024. See *z/VM: CP Programming Services* for more details on this DIAGNOSE code.

Usage Notes

All parameters are ignored by this module.

VSCREEN ALARM

Authorization

General User

Purpose

Use the VSCREEN ALARM command to sound the terminal alarm the next time a virtual screen is displayed in a window on the screen.

Operands

vname
is the name of the virtual screen for which the alarm sounds.

Responses

The terminal alarm sounds.

Messages and Return Codes

- DMS386E Missing operand(s) [RC=24]
- DMS388E Invalid keyword: *keyword* [RC=24]
- DMS391E Unexpected operand(s): *operand* [RC=24]
- DMS921E Virtual screen *vscreen* is not defined [RC=28]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

VSCREEN CLEAR

Authorization

General User

Purpose

Use the VSCREEN CLEAR command to erase data in a virtual screen.

Operands

vname
is the name of the virtual screen to be cleared.

Usage Notes

1. The VSCREEN CLEAR 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
2. 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 VSCREEN DEFINE and SET VSCREEN commands.
3. 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 VSCREEN CLEAR may be different from the cursor line number returned when QUERY CURSOR *vname* is issued immediately following VSCREEN CLEAR. The physical location of the cursor in the virtual screen remains unchanged.

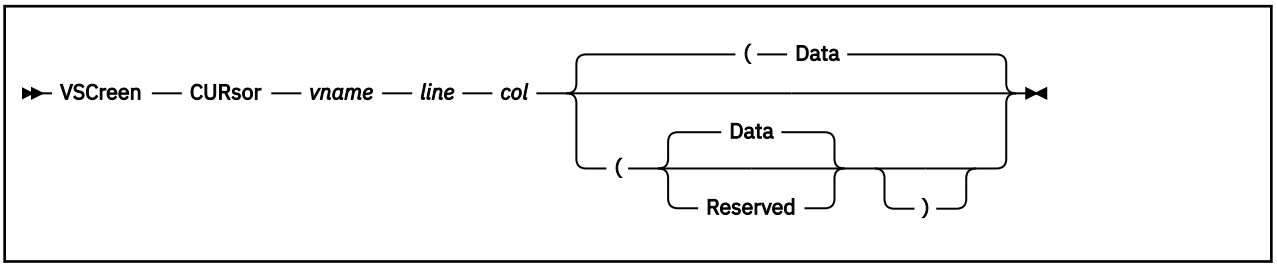
Messages and Return Codes

- DMS386E Missing operand(s) [RC=24]
- DMS388E Invalid keyword: *keyword* [RC=24]
- DMS391E Unexpected operand(s): *operand* [RC=24]
- DMS921E Virtual screen *vname* is not defined [RC=28]
- DMS928E 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 their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

VSCREEN CURSOR



Authorization

General User

Purpose

Use the VSCREEN CURSOR command to position the cursor on a specified line and column in a virtual screen.

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 column number 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. The default is DATA.

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, in column two.

The column number must be less than or equal to the number of columns in the virtual screen, and must be zero or greater. The cursor will not be positioned on the line specified if the line is not within the range of the current top and one line after the current bottom of the virtual screen, or if the line specified is not showing. This allows for a start field in column one. The *start field* (also called *attribute byte*) of a field is a byte that defines field display characteristics like color, highlighting, outlining and so forth.

Usage Notes

Numbering of Reserved Lines

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, and so forth. 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, and so forth.

When the physical screen is read, the following cursor information is saved:

1. Cursor position on the physical screen
2. Cursor position in the window and the window name
3. 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 VSCREEN CURSOR 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.

How the Cursor is Positioned on the Physical Screen

When the physical screen is refreshed, CMS follows the following process to position the cursor on the physical screen:

1. If a border command (except X and H) was entered, the cursor will be positioned on the corner of the window in which it was typed. If that window position is no longer displayed:
2. CMS places the cursor in the vscreen that last owned it.
3. If the last window to own the cursor is connected to this vscreen, CMS checks to see if the window contains the desired vscreen cursor position. If it is visible on the physical screen, the cursor is positioned there.
4. If it is not visible, and the last vscreen to own the cursor was the CMS vscreen, the cursor is placed on the command line.
5. If it was not the CMS vscreen, CMS checks all the other windows connected to this vscreen (starting with the topmost window) to see if the virtual screen cursor is visible on the physical screen. If it is, it is put in the first position found.
6. If no position is found, CMS will check those windows again (starting with the topmost), to see if any are displaying any vscreens which contain any visible cursor positions. The cursor will be placed in the first available position found.
7. If none of the vscreens has any visible positions, CMS will go through all the windows shown on the physical screen (starting with the top-most) to see if the cursor is visible on any of them. If it is, the cursor is placed in that position.

CMS will then try to place the cursor in the *window* that last owned it by following these steps:

1. The cursor will be placed in the window that last owned it if the position is visible on the physical screen. (The position is considered not visible if it has been covered by another window.)
2. If that position is not visible, CMS determines if any position in this window is visible on the screen, and places the cursor in the first visible position found.
3. If no position in this window is visible, CMS will calculate the cursor's position if it were in the top window. If it would be visible there, it is placed there. If not, it is placed in the first visible position of the top window.
4. If the cursor still has not been positioned, it is placed at 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 the location is displayed somewhere in an associated window on the physical screen.

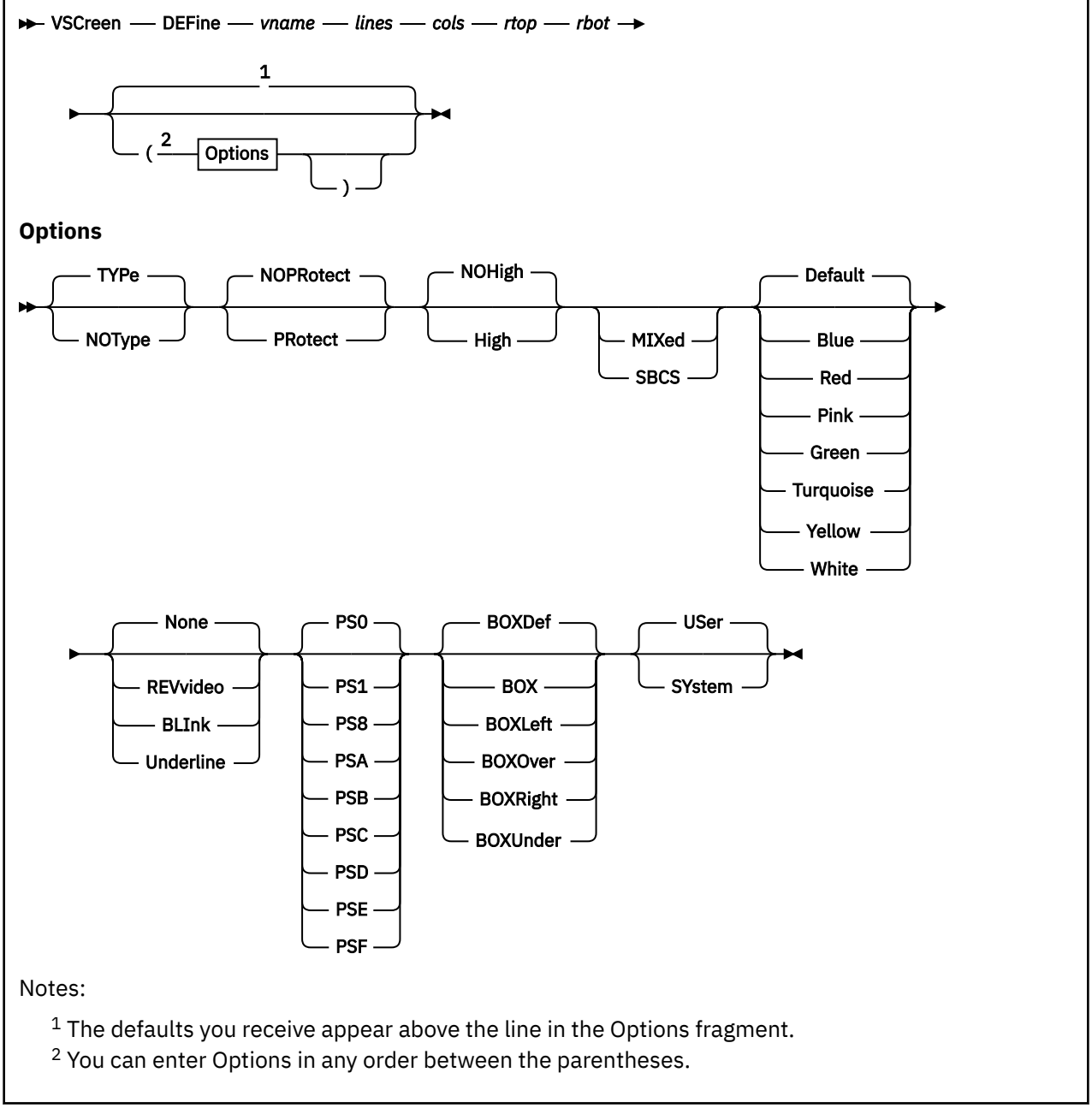
Messages and Return Codes

- DMS386E Missing operand(s) [RC=24]
- DMS388E Invalid keyword: *keyword* [RC=24]
- DMS389E Invalid operand: *operand* [RC=24]
- DMS391E Unexpected operand(s): *operand* [RC=24]
- DMS394E Invalid option: *option* [RC=24]
- DMS921E Virtual screen *vname* is not defined [RC=28]
- DMS923E Specified location is outside the virtual screen [RC=32]
- DMS928E 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 their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

VSCREEN DEFINE



Notes:

[1] The defaults you receive appear above the line in the Options fragment.
[2] You can enter Options in any order between the parentheses.

Authorization

General User

Purpose

Use the VSCREEN DEFINE 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.

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 the virtual screen contains. The number of lines must be one or greater.

cols
is the number of columns 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.

Options

The following may be specified:

TYPe
specifies data is moved to the virtual screen when the virtual screen queue is processed. The default is TYPE.

NOType
specifies the virtual screen is not updated.

PRotect
the data is protected.

NOPRotect
the data is not protected. The default is NOPRotect.

In a NOPROTECT field, all windows are protected *except* the one showing a vscreen from which you are doing a VSCREEN WAITREAD. For more information, see VSCREEN WAITREAD, Usage Note "1" on page 1187.

High
data is displayed in high intensity.

NOHigh
data is displayed in a normal intensity. The default is NOHigh.

MIXed
data is displayed in a mixed DBCS (with SO/SI positions) field.

SBCS
data is displayed in a single-byte character set field.

Default
Blue
Red
Pink
Green
Turquoise
Yellow
White
are the choices for the color of the field.

None
REVvideo
BLInk
Underline
are the choices for the extended highlighting of the field. The default is None.

PS0
PS8
PSA
PSB
PSC
PSD
PSE
PSF
are the choices for the programmed symbol set (PSset) of the field. The default is PS0. PS8 specifies a pure DBCS field.

BOXDef
BOX
BOXLeft
BOXOver
BOXRight
BOXUnder
are the choices for the field outlining for a PS/55-family display. Outlining may be BOXDEF (the default outlining for the device), BOX (a full box), or any combination of the following to obtain the other possible 14 valid values:

BOXLeft
A vertical line on the left of the field

BOXOver
Overline

BOXRight
A vertical line on the right of the field

BOXUnder
Underline

You cannot specify BOX and BOXDef with each other or with any of the above outlining values.

USer
indicates the virtual screen is deleted when a task abnormally ends (abend) or when the HX (halt execution) command is issued. This is the default.

SYstem
indicates 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, see *3270 Data Stream Programmer's Reference*.
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. For more information, see "SET LOGFILE" on page 965.
4. The virtual screen options are accepted whether 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, 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 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
- Outlining

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, as with VSCREEN WRITE and VSCREEN GET, the virtual screen defaults are used unless the write specifies different characteristics, in which case the write options (in VSCREEN WRITE or the LINEWRT macro) override the characteristics of the virtual screen set with VSCREEN DEFINE. For more information, see “VSCREEN WRITE” on page 1192.
7. If MIXED or SBCS is not specified, the default is set based on device characteristics. If the display is a PS/55-family display, the default is MIXED. If the display is not a PS/55-family display, the default is SBCS.
8. The reserved lines are maintained outside the area defined as scrollable data in the virtual screen. For example:

   ```
   vscreen define 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 VSCREEN WRITE command enables you to write to the scrollable data area and, using the RESERVED option, write to the reserved areas.
9. Currently on PS/55-family displays, for fields ending at column 80 of the physical screen that are outlined on the right (with BOX or BOXRight), the outlining on the right actually appears to the left of column 1 on the next line.
10. 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 1-8 characters.
    - The valid characters are A-Z, a-z, 0-9, $, #, @, +, - (hyphen), :(colon), and _ (underscore).
11. To view the contents of a virtual screen in a window, a window must be connected to the virtual screen using the WINDOW SHOW or WINDOW HIDE command.
12. SET CHARMODE must be ON to display characters using programmed symbol set 1 (PS1).
13. Some programs use X'1D' to affect highlighting or color attributes of output. In full-screen CMS X'1D' is a nondisplayable character and does not affect the output.

Messages and Return Codes

- DMS014E Invalid function *function* [RC=24]
- DMS386E Missing operand(s) [RC=24]
- DMS389E Invalid operand: *operand* [RC=24]

- DMS391E Unexpected operand(s): *operand* [RC=24]
- DMS394E Invalid option: *option* [RC=24]
- DMS3952E Conflicting option *option* [RC=24]
- DMS622E Insufficient free storage [RC=104]
- DMS913E Invalid virtual screen name: *vname* [RC=20]
- DMS920E Virtual screen *vname* already exists [RC=3]
- DMS926E 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 their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

VSCREEN DELETE

Authorization

General User

Purpose

Use the VSCREEN DELETE command to remove a virtual screen definition.

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" on page 937).
2. When deleting a virtual screen and data is in the queue, handling of data depends on whether 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 VSCREEN ROUTE command message classes that have been directed to the virtual screen are rerouted to the CMS virtual screen.

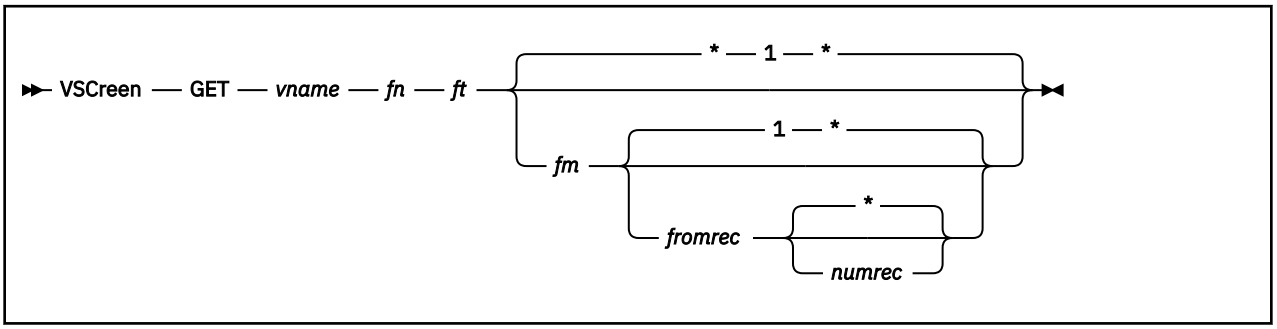
Messages and Return Codes

- DMS386E Missing operand(s) [RC=24]
- DMS388E Invalid keyword: *keyword* [RC=24]
- DMS391E Unexpected operand(s): *operand* [RC=24]
- DMS919E The CMS virtual screen cannot be deleted [RC=24]
- DMS921E Virtual screen *vname* is not defined [RC=28]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	"Command Syntax Error Messages" on page 1411

VSCREEN GET



Authorization

General User

Purpose

Use the VSCREEN GET command to write lines from a CMS file to the specified virtual screen.

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, all accessed disks and directories are searched.

fromrec
is the starting record of the file that is moved into the virtual screen. VSCREEN GET starts with the first record in the file by default.

numrec
is the number of records read from the file and moved into the virtual screen. All the records are read by default. An * means records are processed until the end of the file is reached.

Usage Notes

VSCREEN GET 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. For information on queues see "VSCREEN WRITE" on page 1192.

Messages and Return Codes

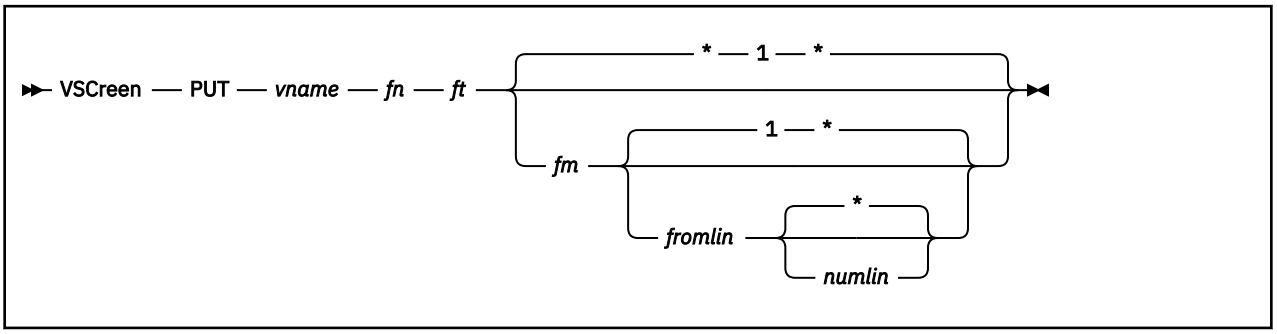
- DMS002E File *fn ft fm* not found [RC=28]
- DMS048E Invalid filemode *mode* [RC=24]
- DMS062E Invalid character *char* in fileid [*fn ft* [*fm*]] [RC=20]
- DMS069E Filemode *mode* not accessed [RC=36]
- DMS104S Error *nn* reading file *fn ft fm* from disk or directory [RC=31 | 55 | 70 | 76 | 99 | 100]

- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS156E Record *nnn* not found -- file *fn ft fm* has only *nnn* records [RC=32]
- DMS386E Missing operand(s) [RC=24]
- DMS388E Invalid keyword: *keyword* [RC=24]
- DMS389E Invalid operand: *operand* [RC=24]
- DMS391E Unexpected operand(s): *operand* [RC=24]
- DMS921E Virtual screen *vname* is not defined [RC=28]
- DMS928E Command is not valid for virtual screen *vname* [RC=12]
- DMS1144E Implicit rollback occurred for work unit *workunitid* [RC=31]
- DMS1252T Rollback unsuccessful for file pool *filepoolid*
- DMS1262S 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 their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411
Errors in the Shared File System	“File Pool Server Messages” on page 1414

VSCREEN PUT



Authorization

General User

Purpose

Use the VSCREEN PUT command to write the data from the scrollable data area of a virtual screen to a CMS file.

Operands

vname
is the name of the virtual screen from which the data is written.

fn
is the file name 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.

fromlin
is the starting line in the virtual screen to be copied into the file. VSCREEN PUT 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.
2. VSCREEN PUT only creates an output file if there are fields defined below the VSCREEN PUT starting line (*fromlin*). Blank lines will be written to the file for each line which does not contain a field. The inverse is not true; VSCREEN PUT only places the fields in an output file up to the last line on which a field is defined. Output may stop before reaching the line specified in *numlin*.

Messages and Return Codes

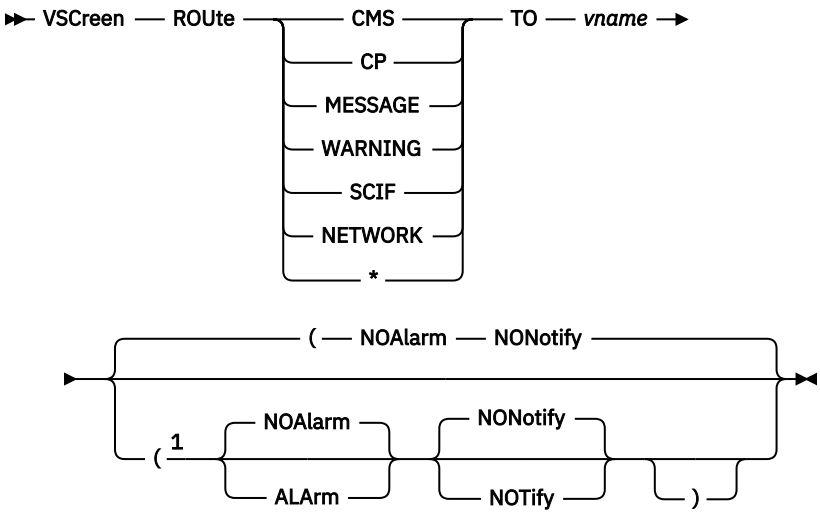
- DMS037E Filemode *mode* is read only [RC=12]

- DMS048E Invalid filemode *mode* [RC=24]
- DMS062E Invalid character *char* in fileid [*fn ft* [*fm*]] [RC=20]
- DMS069E Filemode *mode* not accessed [RC=36]
- DMS105S Error *nn* writing file *fn ft fm* on disk or directory [RC=31 | 55 | 70 | 76 | 99 | 100]
- DMS107S Disk *mode (vdev)* is full [RC=100]
- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS386E Missing operand(s) [RC=24]
- DMS388E Invalid keyword: *keyword* [RC=24]
- DMS389E Invalid operand: *operand* [RC=24]
- DMS391E Unexpected operand(s): *operand* [RC=24]
- DMS531E Disk is full; set new filemode or clear some disk space [RC=13]
- DMS921E Virtual screen *vname* is not defined [RC=28]
- DMS923E Specified location is outside the virtual screen [RC=32]
- DMS924E Data was truncated [RC=3]
- DMS928E Command is not valid for virtual screen *vname* [RC=12]
- DMS936W Virtual screen *vname* is empty [RC=0]
- DMS1144E Implicit rollback occurred for work unit *workunitid* [RC=31]
- DMS1252T Rollback unsuccessful for file pool *filepoolid*
- DMS1258E You are not authorized to write to file *fn ft fm* | *dirid* [RC=28]
- DMS1262S 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 their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411
Errors in the Shared File System	“File Pool Server Messages” on page 1414

VSCREEN ROUTE



Notes:

[1] You can enter Options in any order between the parentheses.

Authorization

General User

Purpose

Use the VSCREEN ROUTE command to direct data of a particular message class to a virtual screen.

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. For more information on the messages classified as NETWORK, see Usage Note “4” on page 1185.

*****
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. The default is NOALARM.

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.

NONotify
will not display the message class name in the status area when you receive a message with a message class of MESSAGE, WARNING, SCIF, or NETWORK. The default is NONOTIFY.

Usage Notes

1. When SET FULLSCREEN is ON, the various message classes are routed to virtual screens as follows:

   Table 56. Initial Settings for Message Routing

Message Class	Virtual Screen	Default 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 applied only to the classes MESSAGE, WARNING, SCIF, and NETWORK. CP and CMS always have the NOALARM and NONOTIFY options.
4. These 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

   116I
   FILE (*orgid*) TRANSFERRED TO *userid*

   147I
   SENT FILE *spoolid*(*orgid*) ON LINK *linkid* TO *nodeid*(*userid2*)

148I
SENT FILE *spoolid* (*orgid*) to partial distribution on link *linkid* to *locid* (*userid*)

170I
FROM *nodeid*: *message-text*

(This message is only considered to be a NETWORK message when *message-text* is a file networking message.)

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.

If an RSCS message is in a language other than American English (AMENG) or uppercase English (UCENG) and does not contain an RSCS message identifier, it will be handled as a regular message.

Messages and Return Codes

- DMS386E Missing operand(s) [RC=24]
- DMS388E Invalid keyword: *keyword* [RC=24]
- DMS391E Unexpected operand(s): *operand* [RC=24]
- DMS394E Invalid option: *option* [RC=24]
- DMS921E Virtual screen *vname* is not defined [RC=28]
- DMS926E Command is only valid in CMS FULLSCREEN mode [RC=88]
- DMS928E 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 their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

VSCREEN WAITREAD

Authorization

General User

Purpose

Use the VSCREEN WAITREAD 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.

Operands

vname
is the name of the virtual screen that is to wait for input.

Usage Notes

1. All windows showing virtual screens other than the virtual screen specified in the VSCREEN 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. VSCREEN WAITREAD 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.
   - The next interrupt is awaited. (The virtual screen specified in the VSCREEN WAITREAD 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 VSCREEN WRITE 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)

   Note:

   a.
   WAITREAD.1 is set to UNKNOWN when the System Request key is used to generate the attention interrupt and no other system. For example, VM/VTAM session managers, and so forth traps the System Request key interrupt.

   b.
   The changed areas (WAITREAD.3, and so forth) may be set when UNKNOWN is returned in WAITREAD.1.

c.
The UNKNOWN value can also be set due to an interrupt by CP. For example, a warning message.

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 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.
specifies 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.
specifies 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.
specifies the cursor was positioned on a line following the bottom of the virtual screen, but 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. VSCREEN WAITREAD 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 (WINDOW DEFINE and VSCREEN DEFINE commands)
   - Connect the window to the virtual screen (WINDOW SHOW command)
   - Write data to the virtual screen (VSCREEN WRITE command)
   - Issue the VSCREEN WAITREAD command.

   When an interrupt is received, the exec can process the WAITREAD.*n* variables and update the virtual screen using the VSCREEN WRITE 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 data in column one is not lost. Lines may be truncated on the right, and the location information indicates 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.
10. If the window or windows showing the active virtual screen are not visible on the screen, the WM window is automatically displayed. See "Things You Should Know about Full-Screen CMS" under SET FULLSCREEN for more information.

Messages and Return Codes

- DMS329W Warning: APL/TEXT option not in effect
- DMS386E Missing operand(s) [RC=24]
- DMS388E Invalid keyword: *keyword* [RC=24]
- DMS391E Unexpected operand(s): *operand* [RC=24]
- DMS494W FULLREAD set off.
- DMS614E Screen modifications lost. See ‘SET FULLREAD’ to use PAkeys safely.
- DMS622E Insufficient free storage [RC=104]
- DMS629W Screen modifications may be lost. Press ENTER key to process screen changes.
- DMS631E WAITREAD can only be executed from an EXEC-2 or REXX exec [RC=4]
- DMS696W Invalid data received from the display
- DMS917E No windows are showing virtual screen *vname* [RC=4]
- DMS921E Virtual screen *vname* is not defined [RC=28]
- DMS925E I/O error on screen [RC=100]
- DMS926E Command is only valid on a display terminal [RC=88]
- DMS928E 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 their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

VSCREEN WAITT

Authorization

General User

Purpose

Use the VSCREEN WAITT command to update a virtual screen with data.

Operands

vname
is the name of the virtual screen to be updated. An * indicates all the virtual screens are updated. An * is the default.

Usage Notes

1. VSCREEN WAITT 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), the data is logged (if logging was requested), and then discarded.
2. Queuing is handled so 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, any messages issued by DMSWVT are usually displayed after the echo and before the other messages.

Messages and Return Codes

- DMS386E Missing operand(s) [RC=24]
- DMS388E Invalid keyword: *keyword* [RC=24]
- DMS391E Unexpected operand(s): *operand* [RC=24]
- DMS921E Virtual screen *vname* is not defined [RC=28]
- DMS928E Command is not valid for virtual screen *vname* [RC=12]

The following messages could be received while logging or updating virtual screen data to disk or directory:

- DMS037E Filemode *fm* is accessed as read/only

- DMS062E Invalid character *char* in fileid *fn ft fm*
- DMS069E Filemode *mode* not accessed
- DMS105S Error *nn* writing file *fn ft fm* on disk or directory
- DMS109S Virtual storage capacity exceeded
- DMS531E Disk or file space is full; set new filemode or clear some space
- DMS924E Data was truncated
- DMS933W Logging stopped for virtual screen *vname*
- DMS934E Text was not written to virtual screen. No field was defined.
- DMS1144E Implicit rollback occurred for work unit *workunitid* [RC=31]
- DMS1252T Rollback unsuccessful for file pool *filepoolid*
- DMS1258E You are not authorized to write to file *fn ft fm* | *dirid*
- DMS1262S Error *nn* opening file *fn ft fm*

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411
Errors in the Shared File System	“File Pool Server Messages” on page 1414

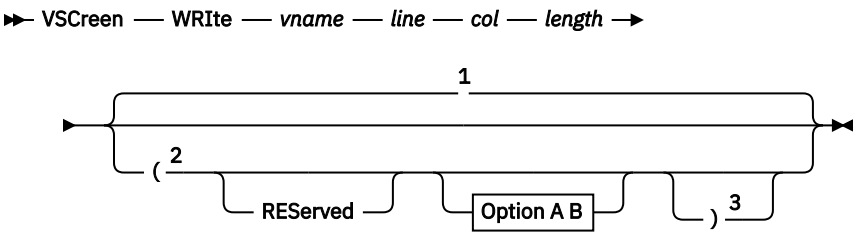
Return codes:

RC
Meaning

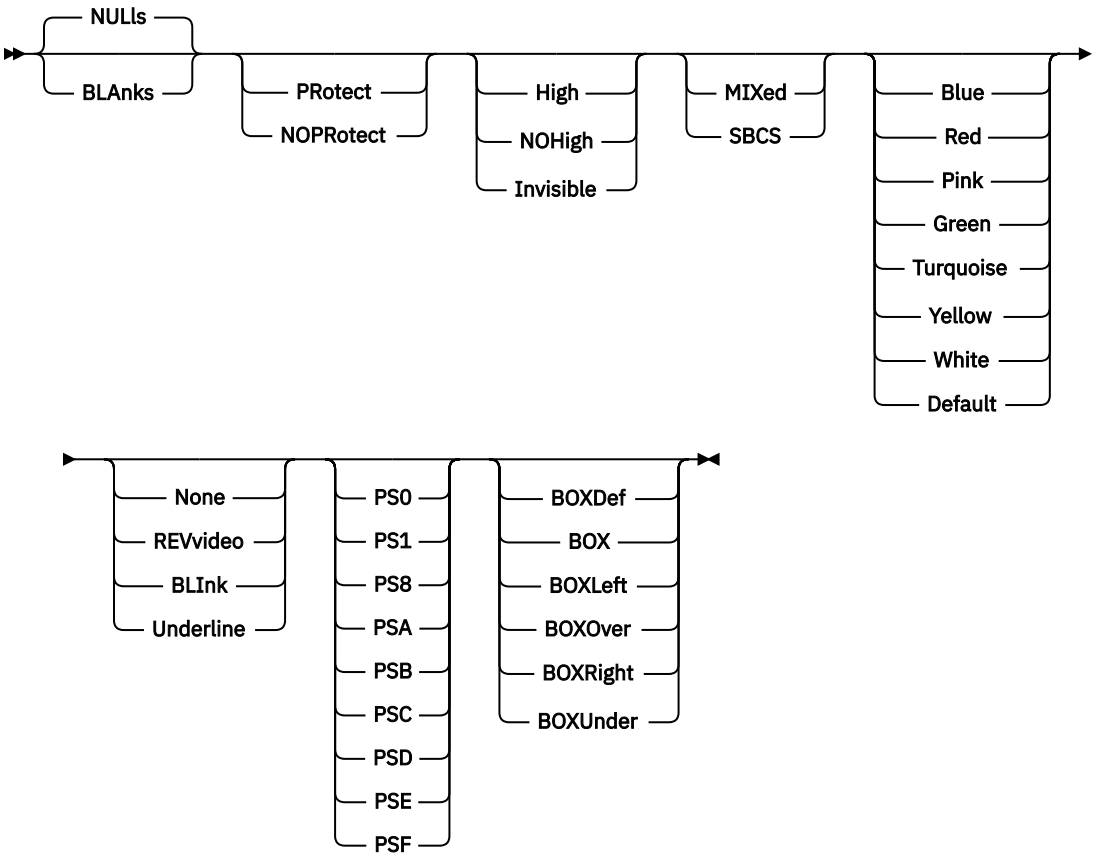
0
Normal execution

13
Virtual screen full

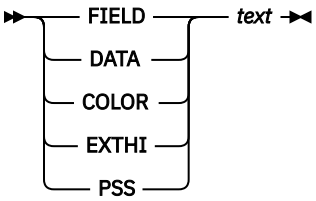
VSCREEN WRITE



Option A



Option B



Notes:

[1] The defaults you receive appear above the line in the Options fragments.
[2] You can enter Options in any order between the parentheses.
[3] If Option B is used, a right parenthesis should not be used to mark the end of the options.

Authorization

General User

Purpose

Use the VSCREEN WRITE command to define fields and modify existing fields in a virtual screen. The information is queued to the virtual screen and is displayed the next time the screen is refreshed.

Operands

vname
specifies the name of the virtual screen.

line
specifies the line number of the virtual screen where the write is to begin.

col
specifies the column number of the virtual screen where the write is to begin.

length
specifies the length to be written.

REServed
specifies 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.

Options

Option A

Option A specifies the padding used for data, field attributes, and extended field attributes. When modifying a previously defined field, any field attribute specified is ignored, while any extended field attribute specified is modified. These may be specified:

BLAnks
pads data with blanks.

NULls
pads data with nulls. This is the default.

PRotect
specifies a protected field.

NOPRotect
specifies a field that is not protected.

High
specifies a high-intensity field.

NOHigh
specifies a normal-intensity field.

Invisible
specifies an invisible field. The data written is not displayed on the screen.

MIXed
defines a mixed DBCS (with SO/SI positions) field when the device is a PS/55-family display.

SBCS
defines a field in which SO and SI codes may not be entered from the keyboard. However, like a MIXED field, data written to this field may contain SO and SI codes.

color
is the color of the field.

exthi
is the extended highlighting of the field.

psset
is the programmed symbol set of the field. It may be PS0, PS1, PS8, PSA, PSB, PSC, PSD, PSE, or PSF. For a loadable psset, the psset must already be loaded in the display. If no psset is loaded, PS0 is used. PS8 specifies a pure DBCS field. SET CHARMODE must be ON to display characters using PS1.

outline
is the field outlining for a PS/55-family display. Outlining may be BOXDef (the default outlining for the device), BOX (a full box), or any combination of the following to obtain the remaining 14 possible valid values:

BOXLeft
A vertical line on the left of the field.

BOXOver
Overline.

BOXRight
A vertical line on the right of the field.

BOXUnder
Underline.

For more information, see "VSCREEN DEFINE" on page 1174.

Option B

Option B specifies the operation to perform. The following may be specified:

FIELD
defines a field, as follows:

- The field will start at the specified line and column (a start field character is placed at this location), and have the length you specify.
- The character attributes of all characters in the field are set to the value which defaults to the extended field attributes.
- When FIELD is specified:
  1. Some of the options in Option A specify the field attribute of the new field. Any of these values not specified will default to the values for the vscreen.
  2. Other options in Option A specify the extended field attributes (color, extended highlighting, programmed symbol set, and field outlining.) Any of these values not specified will default to the values for the vscreen.
  3. Specify the data for the field in option B. The data will begin at the byte immediately following the start field character, and will be padded (as specified in option A) or truncated to the end of the field. The field can only hold one less character than the length you specify, due to the existence of the start field character.

DATA
modifies the data in a previously defined field. The character attributes may also be modified.

- The modified portion of the field starts at the specified line and column and continues for the length you specify. If that length is greater than the size of the field, it will be truncated to the end of the field.
- When DATA is specified:
  1. You will also specify the new data for the modified portion of the field with option B. It will be padded (as specified in option A) or truncated to the end of the modified part of the field.
  2. Some of the options in Option A will specify the new character attributes (color, extended highlighting, programmed symbol set, and field outlining) of all characters in the modified part of the field. Any of these values not specified will not be changed.
  3. The field attribute is not changed.

4. The extended field attributes are not changed.

COLOR
modifies the color character attributes in a previously defined field. The data and the other character attributes (extended highlighting and programmed symbol set) may also be modified.

- The modified portion of the field starts at the specified line and column and continues for the length you specify. If that length is greater than the size of the field, it will be truncated to the end of the field.
- When COLOR is specified:

  1. Option B will specify the new color character attribute for each character in the modified part of the field. 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

  The text will be padded or truncated to the end of the modified part of the field. If padding occurs, it will be with the color value specified in option A, if one is given; otherwise, padding will be with the vscreen color.
- Extended highlighting or programmed symbol set values in option A specify the new character attributes of all characters in the modified portion of the field. Any option A values not specified will not be changed.
- Option A specifies the new data for the modified portion of the field. If option A is not specified, the data will not be modified.
- The field attribute is not changed.
- The extended field attributes are not changed.

EXTHI
modifies the extended highlighting attributes in a previously defined field. The data and the other character attributes (color and programmed symbol set) may also be modified.

- The modified portion of the field starts at the specified line and column and continues for the length you specify. If that length is greater than the size of the field, it will be truncated to the end of the field.
- When EXTHI is specified:

  1. Option B will specify the new extended highlighting attribute for each character in the modified part of the field. The following characters represent the extended highlighting:

     0
     Defaults to the extended highlight of the field

     1
     Blink

2
Revvideo

4
Underline

The text will be padded or truncated to the end of the modified part of the field. If padding occurs, it will be with the extended highlighting value specified in option A, if one is given; otherwise, padding will be with the vscreen extended highlighting.

2. Color or programmed symbol set values in option A specify the new character attributes of all characters in the modified portion of the field. Any option A values not specified will not be changed.
3. Option A specifies the new data for the modified portion of the field. If option A is not specified, the data will not be modified.
4. The field attribute is not changed.
5. The extended field attributes are not changed.

PSS
modifies the programmed symbol set attribute in a previously defined field. The data and the other character attributes (color and extended highlighting) may also be modified.

- The modified portion of the field starts at the specified line and column and continues for the length you specify. If that length is greater than the size of the field, it will be truncated to the end of the field.
- When PSS is specified:

  1. Option B will specify the new programmed symbol set attribute for each character in the modified part of the field. The following characters represent the PSS: 0 and blank (default to the PSS of the field if the PSS is a loadable PSS, default to PS0, which is the terminal default, if the PSS is non-loadable), 1, A, B, C, D, E, F,

     Note: The letters must be specified in upper case.

     The text will be padded or truncated to the end of the modified part of the field. If padding occurs, it will be with the programmed symbol set value specified in option A if one is given; otherwise, padding will be with the vscreen programmed symbol set value.
  2. Color or extended highlighting values in option A specify the new character attributes of all characters in the modified portion of the field. Any option A values not specified will not be changed.
  3. Option A specifies the new data for the modified portion of the field. If option A is not specified, the data will not be modified.
  4. The field attribute is not changed.
  5. The extended field attributes are not changed.

text
is the data to be written in the field. Mixed DBCS data can be used for *text* when *text* is preceded by the FIELD or DATA keyword, and provided your terminal is capable of supporting mixed DBCS.

Usage Notes

1. When VSCREEN WRITE is issued, the text and its characteristics are held in the virtual screen queue created by VSCREEN DEFINE. To move the text from the queue to the virtual screen, you should issue either VSCREEN WAITREAD, VSCREEN WAITT, or the PSCREEN REFRESH command. This process is required when issuing the VSCREEN WRITE command from an EXEC.
2. Use the INVISIBLE option to prevent data from being displayed on the screen. The INVISIBLE option causes the text of the field to be ignored. A field of blanks or nulls will be defined for the length specified. You should use the LINERD macro with the INVISIBLE option for sensitive data, such as passwords.

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 B is not specified, the default is FIELD.

   For more information on fields, see *3270 Data Stream Programmer's Reference*.

4. 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, no start field is inserted, and the text begins in the column specified, (or in the next available column if that location is a start field).

5. Working with DBCS

   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.

6. PS8 defines a pure DBCS field in a vscreen, when the device is a PS/55-family display with PS8 capability. You can define a pure DBCS field only when you define a field (VSCREEN WRITE with FIELD option). Once you specify PS8, you cannot change to another *psset* value without redefining the field. If you specify both PS8 and MIXED or SBCS, PS8 takes precedence and MIXED or SBCS is ignored.

7. Writing Pure DBCS Data

   After a pure DBCS field is defined, you can use the DATA option to change the DBCS string in the vscreen data buffer. You can only use the DATA option to write pure DBCS text into a field already defined as pure DBCS. If you have specified PSS for the *text* in option B, PS8 is ignored if specified in option A.

   If you are writing to a pure DBCS field that wraps lines such that the PS8 data in the line ends up being an odd number of bytes in length, blanks will be inserted at the beginning or end of the line.

8. All DBCS characters in the text written with VSCREEN WRITE with the FIELD or DATA option are checked for validity. The characters that are not valid are replaced by the NONDISP character defined for CMS, or by characters that indicate there is no recognizable representation for the alphanumeric characters being written.

9. Alignment of Pure DBCS Data

   Data written in a predefined pure DBCS field must begin at a column that is the first byte of a DBCS character. If it does not do so, the text will be aligned this way, and no message or nonzero return code will be issued.

   For example, suppose you have a field at line 8 of your MESSAGE vscreen that contains a DBCS string, where @ represents the start field, K represents the first byte and 1-9 the second byte of a valid DBCS character in hexadecimal, as follows:

   ```
   @K1K2K3K4
   ```

   You want to write one DBCS character of text, K9, at column 5 (the *2* of K2) of this field. If your program issues:

   ```
   VSCREEN WRITE MESSAGE 8 5 2 (PS8 DATA K9
   ```

   You will get the following string as a result:

   ```
   @K1K9K3K4
   ```

   K9 is automatically aligned so it starts at the fourth column rather than the fifth, because starting it at the fifth column would have resulted in a not valid DBCS string:

   ```
   @K1KK93K4
   ```

10. Truncation of Pure DBCS Data

- Truncation of the Field

  Because a pure DBCS field holds 2-byte DBCS characters of text, plus a 1-byte start field, the field length must be odd. If it is an even number of bytes, it is shortened by one byte.

  For example, if your program issues:

  ```
  VSCREEN WRITE MESSAGE 8 3 6 (PS8 FIELD K1K2K3K4
  ```

  (where 6 is the field length), it will be processed as:

  ```
  VSCREEN WRITE MESSAGE 8 3 5 (PS8 FIELD K1K2K3K4
  ```

- Truncation of the Text

  If the field is truncated and becomes shorter than the text you are writing to it, the text is also truncated to fit the field and a message is issued. For example, the above commands write the following field:

  ```
  @K1K2
  ```

  which has a length of five. The original data, K1K2K3K4, has been truncated to fit the field.

- Truncation of not valid DBCS Data

  When defining a PS8 field (FIELD option) or writing to a predefined PS8 field (DATA option), the text must also be an even number of bytes. If it is an odd number of bytes, it will be shortened by one byte to make it valid, with no message issued.

  For example, if your program issues:

  ```
  VSCREEN WRITE MESSAGE 8 3 0 (PS8 FIELD K1K2K
  ```

  (where K1K2K is the text), it will be processed as:

  ```
  VSCREEN WRITE MESSAGE 8 3 0 (PS8 FIELD K1K2
  ```

11. When MIXED or SBCS is specified, both DBCS and SBCS characters may be displayed in the field, separated by 1-byte SO and SI control codes. You may insert and type over SO/SI codes in a MIXED field, but you cannot enter SO/SI codes directly from the keyboard in a SBCS field.

    If your application writes output to a mixed DBCS field that causes the field to wrap a line, an extra start field or extra SO/SI characters may be inserted on the following line(s) for the text that wraps. Allow extra space for these when you specify the field length; otherwise, truncation may occur. If you overtype a wrapped field, the extra characters are returned to the application in the changed field.

    Once you define a MIXED or SBCS field, you cannot change the field attributes without redefining the field.

    The MIXED and SBCS options are ignored if the PS8 option is also specified. If you are writing a MIXED or SBCS field on a PS8 vscreen, be sure to specify PS0 or the PSS value you want. Otherwise, PS8 is assumed, and MIXED or SBCS is ignored.

12. Field Outlining

    You can define field outlining only when you define a field (using the FIELD option).

    Outlining may be BOXDef (the default outlining for the device), BOX (a full box), or any combination of the following to obtain the remaining 14 possible valid values:

    BOXLeft
    A vertical line on the left of the field

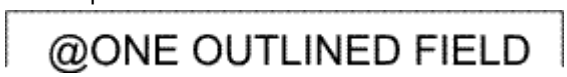
    BOXOver
    Overline

    BOXRight
    A vertical line on the right of the field

BOXUnder
Underline

You cannot specify BOX and BOXDef with each other or with any of the other outlining values above.

If a field is broken or wraps around a line, the outlining is the same for each partial field or line as it was for the original field. For example:



If the above field is broken into two fields by a second field being defined, the outlining (BOXRight, BOXOver, and BOXLeft) repeats itself for the second field, as follows:

If the field wraps around a line, the outlining (BOXRight, BOXOver, and BOXLeft) repeats on each line:

Vertical outlining that wraps lines, as in this example, may produce the appearance of multiple fields on the screen, although only a single field is being displayed.

Note: Currently on PS/55-family displays, for fields ending at column 80 of the physical screen that are outlined on the right (with BOX or BOXRight), the outlining on the right actually appears to the left of column 1 on the next line.

13. When you enter the VSCREEN WRITE command and logging is in effect (SET LOGFILE command), CMS converts pure DBCS to mixed DBCS text (by adding SO/SI control codes) before putting the data into a file. This is done so you can use XEDIT to view or change the file.

    Logging does not take place until a refresh occurs.

14. 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:

      ```
      vscreen write cms 0 0 0 (field Enter 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:

      ```
      vscreen write 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:

      ```
      vscreen write 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.

15. When using Option B, 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 you intended only for a character.
16. 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:

    ```
    vscreen write cms 0 5 0 (COLOR 11116666777
    ```

    will update the color buffer starting at the fifth position of the current field.
17. If the length specified is less than the length of the text, the text is truncated. If the length specified is greater than the length of the text, the text is padded with the characteristics specified in option A. If the options are not specified, text is padded with the characteristics defined for the virtual screen.

    Note: When you are not writing a field, the text is written for the length specified or until the next field is encountered.

    For example:

    ```
    vscreen write cms 1 1 20 (blank field Enter your name:
    vscreen write cms 0 0 20 (red color 11111
    ```

    writes *Enter your name:*. The *Enter* is displayed in blue and *your name:* is displayed in red.

    The example:

    ```
    vscreen write cms 1 1 10 (data Enter your name:
    ```

    writes *Enter your*.
18. 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.
19. 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, and so forth. 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, and so forth.
20. 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, psset) or data of the field. For example, suppose you define the this field:

    ```
    vscreen write cms 5 20 0 (field Enter your name:
    ```

    To change the color extended attribute, you can enter:

    ```
    vscreen write cms 5 20 0 (COLOR 2222224444455555
    ```

    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.
21. When writing sequentially, see Usage Note “14” on page 1199, and the text contains a character assigned to X'15' (end of line, or EOL) through 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.
22. You should load programmed symbol sets prior to using XEDIT or prior to issuing the SET FULLSCREEN or SET FULLSCREEN RESUME commands. This ensures 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.

23. Some programs use X'1D' to effect highlighting or color attributes of output. In full-screen CMS X'1D' is a nondisplayable character and does not affect the output.

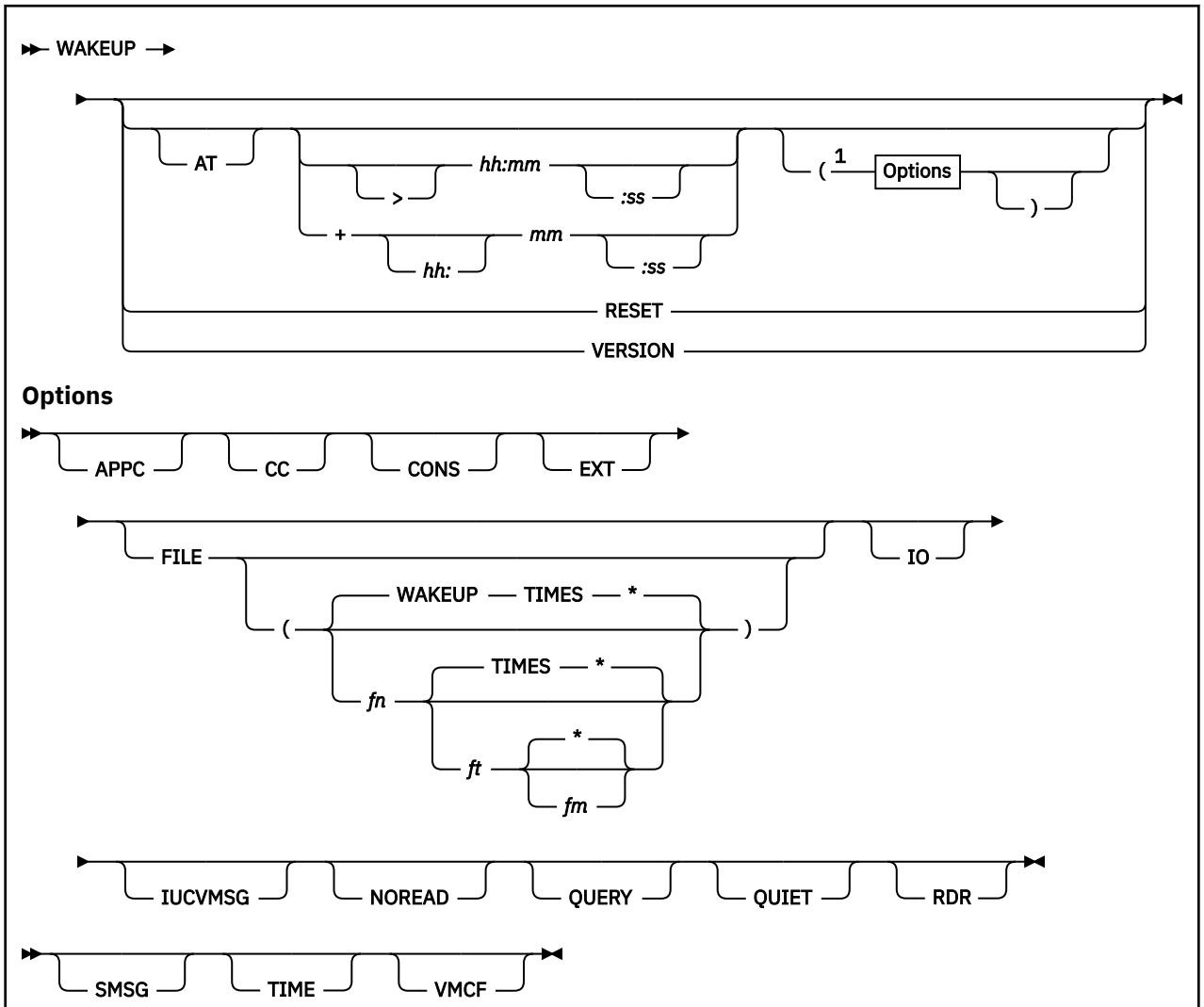
Messages and Return Codes

- DMS622E Insufficient free storage [RC=104]
- DMS921E Virtual screen *vname* is not defined [RC=28]
- DMS923E Specified location is outside the virtual screen [RC=32]
- DMS928E Command is not valid for virtual screen *vname* [RC=12]
- DMS3952E Conflicting option *option* [RC=24]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

WAKEUP



Notes:

[1] You can enter Options in any order between the parentheses.

Authorization

General User

Purpose

The WAKEUP command controls the startup of an event-driven virtual machine (typically a disconnected service virtual machine) by ending its wait state condition whenever a specified event occurs. WAKEUP events can be specified using optional WAKEUP parameters. WAKEUP events that may end a wait state include:

- The passing of a time of day (including a date or day of the week)
- The presence or arrival of reader files
- The arrival of a Virtual Machine Communication Facility message
- The arrival of a Special Message Facility message (from CP SMSG).

The return code from WAKEUP indicates which WAKEUP event occurred.

Note: You may use WAKEUP in an environment above 16MB; however, it will be loaded below 16MB.

Operands

>*hh*:*mm*:(*ss*)
refers to the time of day you want WAKEUP to restart the virtual machine. The format of this parameter is fixed, but the seconds (*:ss*) are optional, with `:00` being their default value. If the restart event is time-of-day, the return code is 2. But the WAKEUP time (and its return code) may be overridden if another restart event occurs first.

Specifying > *hh*:*mm*(:*ss*) causes WAKEUP to return right away if it is already past that time. For example, if you specify:

```
wakeup >14:30
```

and it is 14:45 (2:45 p.m.), WAKEUP will return immediately. (It will not wait until 14:45 the next day.) If you do use the > before the time, you can only specify the time to the most recent 10-second interval (for example, both `>02:22:2` and `>02:22:28` mean 2:22:20 a.m.).

If you do not specify > before the time, and another 60 seconds go by, WAKEUP will wait until the specified time *the next day* to restart. For example, if you entered the following:

```
wakeup at 08:15:10
```

And, if you entered the record *before* 8:15:10, WAKEUP will restart the virtual machine at 8:15:10 a.m. If you entered the record between 8:15:10 and 8:16:09, WAKEUP will restart it *immediately*. And if you entered the record after 8:16:09, WAKEUP will restart it at 8:15:10 a.m. *the next day*.

+(*hh*:)*mm*:(*ss*)
specifies a time interval from *right now* for the virtual machine to be restarted. To work, the parameter must begin with a *+* sign. Hours (*hh:*) and seconds (*:ss*) are optional, with `00:00` being their default values. Some valid forms are:

+0
"Wake up *now*."

Use this to check for one or more WAKEUP events (options) without having to wait for them to occur. For example,

```
wakeup +0 (rdr iucvmsg
```

returns RC=4 if a reader file is found, RC=5 if a message is waiting, or RC=2 otherwise. When you specify +0, WAKEUP will not wait—even if it doesn't find a reader file or a message already there.

+10
"Wake up 10 minutes from now."

+30:30
"Wake up 30 minutes and 30 seconds from now."

+02:30:0
"Wake up 2 hours and 30 minutes from now."

If you specify the restart event as a time interval from right now, the return code is 2. But this time interval (and its return code) may be overridden if another restart event occurs first.

If you use the *+hh:mm:ss* form, you can only specify the time to the closest 10–second interval (for example, both +02:30:2 and +02:30:28 mean plus 2 hours, 30 minutes, and 20 seconds).

RESET
terminates Inter-User Communication Vehicle (IUCV) and Virtual Machine Communication Facility (VMCF) communications. Do not specify any other operands or options with RESET.

VERSION
returns WAKEUP's version number in the return code. For example,

```
wakeup version
Ready(00010);
```

indicates WAKEUP version 1.0.

Do not specify any other operands or options with VERSION.

Options

APPC
does a SET SERVER ON for you. It must be used with the EXT option. Because it is an EXTernal interrupt, it will return a RC 8 just as the EXT option does. If you specify APPC with the EXT option and an interrupt occurs, then WAKEUP returns with RC=8, and the following line on the stack:

```
    *EXT interruption-code
```

If you specify APPC without the EXT option, you will receive message DMSCYW2247W and the APPC option will be ignored.

CC
the clock comparator timer is the only timer used by WAKEUP.

CONS
waits for an attention interrupt from the virtual console (caused by typing something and pressing ENTER, or by pressing ENTER twice without typing anything). With CONS it's easy to end a wait state at any time. The return code is set to 6.

Note: The CONS option is implied for any WAKEUP option that waits for some event. Specifying CONS explicitly will cause WAKEUP to exit immediately if a line is already on the stack. This prevents an attention interrupt from being lost if the user types something between a stack check in the exec and WAKEUP invocation.

EXT
waits for any external interrupt.

EXT does the same thing for external interrupts that the IO option does for input/output interrupts. If you specify EXT and an external interrupt occurs, then WAKEUP returns with RC=8 and the following line on the stack:

```
    *EXT interruption-code
```

The EXT option lets WAKEUP be used together with programs using VMCF or IUCV, in a more sophisticated manner than WAKEUP supports by itself. Many programs that need better support for these facilities still find WAKEUP useful for processing timer events and reader files.

By specifying EXT, WAKEUP simply notes when an external interrupt occurs. It still invokes the normal procedure for processing external interrupts, so that a program that has established an external interrupt handler, through the HNDEXT macro or by stealing the EXTNPSW, will still get control. After that program has finished processing the interrupt, WAKEUP will return to the caller with the interruption code (for example, 4001 for VMCF, 4000 for IUCV) on the stack. The calling program can then invoke another module to properly process the interrupt.

If another option which uses external interrupts is specified on the WAKEUP command, then external interrupts generated by that option will be processed by that option, not by the EXT option. So if you specified:

```
    wakeup (iucvmsg ext
```

IUCV interrupts would be processed by returning RC=5, but VMCF and SMSG interrupts would be processed through EXT. (Of course, for that to happen you would have had to invoke a program to enable VMCF before calling WAKEUP.)

FILE (*fn*(*ft*(*fm*)))
causes WAKEUP to search the specified file for the next WAKEUP time. If this option is specified without parameters, the FILE option causes WAKEUP to search the WAKEUP TIMES * file for the next WAKEUP time.

Your WAKEUP TIMES file may be fixed or variable length with a maximum logical record length (LRECL) of 247.

WAKEUP examines each record in the file, using these criteria:

1. Skip the record if it should not be executed today.
2. Skip the record if it has already been executed today and it is not in a relative time interval (+*mm*).
3. The record selected for WAKEUP must pass the two tests above and have the earliest scheduled execution time.

If the record that was selected has a time earlier than the current time, WAKEUP stacks the record and returns immediately. If the time has not passed, WAKEUP compares the WAKEUP time on the record with the WAKEUP time specified as a parameter to the WAKEUP command, and waits until the shortest of the two intervals has passed before returning.

In other words, WAKEUP executes the records in the file of WAKEUP TIMES in the correct temporal order, regardless of their order in the file. However, sometimes the records may be executed late (for example, if a record's time was 1 a.m., but the system was not IPLed until 6 a.m.).

To make WAKEUP sleep all day, add a statement similar to this to your file of WAKEUP TIMES:

```
ALL 23:59:59 01/01/01
```

Then WAKEUP will always have a reason to wait. When WAKEUP wakes up for this event, you could have your exec sleep for 1 minute. At the end of the 1 minute it will be the next day. When your loop re-executes and WAKEUP is reinvoked, the WAKEUP TIMES file will be read for the next day. Otherwise, WAKEUP will return when there are not any more tasks for today and will not sleep until tomorrow.

To reinvoke WAKEUP at midnight, be sure to use either a relative time on the command or a relative time in the file, or have a time near midnight in the file that will cause your application to wait until midnight has passed before reinvoking WAKEUP. Specify midnight as `00:00` to cause WAKEUP to wake up at the start of the day.

A sample file is provided called WAKEUP SAMPLE, which you can tailor for your use. WAKEUP SAMPLE has

```
ALL    23:58:00  01/01/01
```

as the last entry. Adding `MSG01` to this entry

```
ALL    23:58:00  01/01/01   MSG01
```

will cause the sample EXEC in Figure 68 on page 1206 to use this entry to sleep through midnight and begin processing again the next day. A similar entry is required for every CMS function wishing to use WAKEUP TIMES to schedule events after midnight. The sample EXEC uses the keyword MSG01 to trigger the exec to sleep through midnight. See the sample EXEC in Figure 68 on page 1206 for details.

To keep track of which records have been executed, WAKEUP time- or date-stamps an 8-byte field following the *hh:mm:ss* field with the time or date the record was last executed. This means that your file of WAKEUP TIMES *must* be on a R/W minidisk or directory. Give your WAKEUP TIMES file a file mode of 6, unless there's a special reason not to. Because mode 6 files are updated in place, making

the file of WAKEUP TIMES file mode 6 ensures that it will always be up-to-date. See Items Stacked by WAKEUP for the format of the file entries.

When a file item is the cause of the WAKEUP, the line from the file is stacked, and WAKEUP exits with the return code set to 3.

When FILE is specified, WAKEUP will always stack the line from the file (or a null line if no line was found) before exiting, even if the cause of the WAKEUP was not timer expiration. The stacked line has the following format:

```
   * recno record-from-file
```

where *recno* is the record number of the line in the file, and *record-from-file* is as much of the record as will fit on the stack.

Figure 68 on page 1206 is a sample application of the WAKEUP FILE option.

```
/* This is a sample application of the WAKEUP 'FILE' option.   */
/* This program uses the sample WAKEUP TIMES file.             */

Address COMMAND
Do forever
  'MAKEBUF'
  'WAKEUP (FILE(WAKEUP)'
  if rc=3 then Do
    pull var1
    'DROPBUF'
    /* parse field 4 from the stacked wakeup times file line   */
    parse upper value var1 with asterisk reqno field1 field2 ,
                                field3 command
    if command='MSG01' then Do
      'CP MSG OPERATOR THE TIME IS NOW:' time() 'ON' date()
      'CP SLEEP 3 MIN'             /* sleep through midnight */
      END
    else
      if command><'' then Do
        if subword(command,1,1)='CMS' then
        command=subword(command,2) /* strip off cms part     */
        address CMS command        /* execute command        */
      end                          /* end of if command><''  */
  end
  else Do
    'DROPBUF'
    leave
  end                                /* end of else Do         */
end                                  /* end of Do forever loop */
exit
```

Figure 68. Sample Exec Using the WAKEUP FILE options

IO

waits for an I/O interrupt from any device.

When the interrupt occurs, WAKEUP returns with RC=7 and the line

```
   *IO devaddr CSW: csw1 csw2
```

on the stack. Using the device address and the channel status word (CSW), you can determine what device caused the interrupt, and why. For example, if the user pressed a PF key, the first byte of *csw2* would be hex 80 (attn); if the user had just dialed your user ID, it would be hex 40 (device end).

WAKEUP only looks for I/O interrupts while it is running. If the device generates an interrupt while WAKEUP is not running (while you are in some other part of your exec, for example), WAKEUP will not return the interrupt to you. For that reason, you may want to poll all the devices periodically to ensure that they are not left waiting forever.

WAKEUP does not interfere with other programs that may also be waiting to process interrupts. Before returning control to the caller, WAKEUP invokes the CMS I/O interrupt handler, which will give

control to another user-defined device interrupt handler, if one has been established. When the other interrupt handler has finished, WAKEUP returns control to the caller.

Console interrupts are never returned via RC=7 and *IO; they always come as RC=6.

If the RDR option is specified, then a file arriving in the reader will still generate the RC=4 response. However, if RDR has not been specified, the device address and CSW for the unit record card reader will be returned, and the return code will be 7.

IUCVMSG
traps arriving messages with the Inter-User Communication Vehicle (IUCV).

You can use the IUCVMSG option to:

1. Create "robot execs" to tell people you are out when they send you messages and you are not at your terminal.
2. Create "service virtual machines" that handle commands and queries sent by messages.[1]
3. Intercept CP/CMS output created by other CP/CMS commands.

Typically, the following sequence should be used in an exec that will be trapping messages with WAKEUP. This sample exec will send a message to the user ID OPERATOR when it starts up to say that it is running. It will take input from another user ID, and then send the text of that message to whatever user ID is requested. For example, if the user ID running this sample exec is user ID SERVER1, then user ID MYUSER might send a message to SERVER1 and have it send a message to user ID YOURUSER.

```
tell server1 youruser Hi, see you at lunch at 11
```

On YOURUSER you will see:

```
* MSG FROM MYUSER: Hi, see you at lunch at 11
```

[1] The SMSG option can also be used to implement a service virtual machine, but its use is restricted to users on the same node. RSCS only sends messages across the net, not SMSGs, so if some users of the service virtual machine are remote, IUCVMSG must be used.

```
/* Invoke WAKEUP first so it will be ready to receive msgs */
/* This call also issues a 'SET MSG IUCV' command.         */
Trace 'Off'

'Dropbuf 0'
"WAKEUP +0 (IUCVMSG"

/* Next, send out queries to other machines, or messages   */
/* to users so they know your exec is ready to go.         */

call alert;

/* In this loop, we wait for a message to arrive and       */
/* process it when it does.  If the "operator" types on    */
/* the console (rc=6) then leave the exec.                 */

Do forever;
   'Makebuf'                /* Get a buffer level         */
   'wakeup (iucvmsg'        /* wait for a message         */
   if rc /=5 then leave;    /* Leave when its not a msg   */
   parse pull type uid text  /* get the msg details...    */
   'Dropbuf'                /* Drop the buffer            */
   call domsg uid,text      /* go process the command     */
end;

/* when its time to quit, come here                        */
xit:

'WAKEUP RESET';        /* turn messages from IUCV to ON    */
exit;

Alert: procedure
  Tell OPERATOR at GDLVM7 'WAKEUP server machine is now up and
                           running'
return

Domsg: procedure expose uid text
if Wordpos('From',text) = 1 then
   parse var text wordfrom usernode text
Tell text
return
```

Figure 69. How an Exec Can Use WAKEUP to Trap Messages

As Figure 69 on page 1208 suggests, when WAKEUP traps a message, it sets a return code of 5 and puts a line on the stack. The line includes the message type (see Figure 70 on page 1209), the user ID of the sender,[2] and the message text itself. Each call to WAKEUP results in just one message being stacked—even if there are several messages outstanding. Just invoke WAKEUP repeatedly to get the rest of the events.

You can cause WAKEUP to intercept warning messages as well as regular messages by issuing the command:

```
   cp set wng iucv
```

And you can temporarily suspend message interception by issuing the commands:

```
   cp set wng on
```

and

```
   cp set msg on
```

Besides messages and warnings, EMSG, IMSG, CPCONIO, VMCONIO, SMSGs, and secondary console image facility terminal I/O can also be intercepted by issuing the appropriate CP SET *xxxx* IUCV

[2] If the sender is not on your node, then the user ID will be the user ID of the network machine, and you will have to parse the actual user ID from the message text yourself.

command. (See the *z/VM: CP Commands and Utilities Reference* for more information about the CP SET command and its EMSG, IMSG, CPCONIO, VMCONIO, and SMSG parameters.)

The following message types are returned as part of the stacked line:

```
*IUCV   - Unknown IUCV message type
*MSG    - CP MSG or MSGNOH command
*WNG    - CP WNG command
*EMSG   - SET EMSG IUCV
*IMSG   - SET IMSG IUCV
*SMSG   - SET SMSG IUCV
*CP     - SET CPCONIO IUCV
*VM     - SET VMCONIO IUCV
*SCIF   - CP MSG or MSGNOH command
```

Figure 70. Returned Message Types

NOREAD
suppresses a read when a console attention interrupt occurs.

Normally, when you cause an interrupt while WAKEUP is running, the terminal is placed in VM READ state and the line you entered is placed on the stack. With full screen applications this is not always desirable.

NOREAD will cause WAKEUP to return with a return code of 6 (indicating that an attention interrupt has occurred), but it is up to the caller to read the terminal. Failure to do so may cause the terminal to enter the NOT ACCEPTED state. To clear this, enter:

```
   #cp attn
```

QUERY
stacks the current settings of the IUCVMSG option. The following stacked lines have the indicated meanings:

```
*QUERY               - IUCVMSG is not on.
*QUERY IUCVMSG       - The IUCVMSG option is on.
```

Figure 71. Current Settings of the IUCVMSG Option

QUIET
suppresses typing of informational messages. This option prevents WAKEUP from typing an event description on the terminal.

RDR
waits for a file in the virtual reader. The file must be of the same class as the spool class of the reader. (Use CP SPOOL RDR CLASS * to wake up for files of any class.)

If a file of the correct class is already present in the reader and the file is not in HOLD status, the module will exit immediately with a return code of 4. However, WAKEUP will not exit for dump files (class dmp) or monitor files (class mon) that are already present in the reader.

The default reader at address 000C must be defined for correct WAKEUP operation with the RDR option. If there is no reader defined at address 000C, message DMSCYW205W will be received and WAKEUP will end with a return code of 201.

A reader at an address lower than 000C can be defined in conjunction with the 000C reader. WAKEUP with the RDR option will not wake up for interrupts on a reader device address other than 000C. This method could be used to filter out reader interrupts. For example, if the lower address reader is spooled CLass A NOHOLD and the default reader is spooled CLass * NOHOLD, WAKEUP will not wake up for class A files that arrive. However, class A files that are already in the reader when WAKEUP is invoked will cause a return code of 4 because the default reader is spooled CLass * NOHOLD.

SMSG
waits for an SMSG or VMCF SENDX transmission. Any form of data transfer other than SMSG or SENDX is rejected. The data received is typed on the console, and is also stacked in one of these forms (noting how the message was sent):

```
*VMCF userid message-text
```

or

```
*SMSG userid message-text
```

Control is then returned to CMS (or to a calling exec) with the return code set to 1.

When this option is given, WAKEUP will do a VMCF AUTHORIZE for any user ID. The maximum data that WAKEUP can receive through SMSG or SENDX is 280 bytes.

The SMSG option implies the VMCF option, so WAKEUP will enable VMCF and issue the command SET SMSG ON.

Note: WAKEUP expects to be the only application waiting for SMSG or VMCF communications in the virtual machine. If another application also enables for SMSG or VMCF, unpredictable results may occur.

SMSGs are queued by CP so that SMSGs sent by other users will not be lost while the machine is executing modules other than WAKEUP.

Note: WAKEUP expects to be the only application waiting for VMCF communications in the virtual machine. If another application also enables for VMCF communications, unpredictable results may occur.

TIME
causes the current date and time to be stacked on exit. If this is the only option, WAKEUP exits with the return code set to 0. The stacked line has the format:

```
* mm/dd/yy hh:mm:ss
```

Because there are now better ways to request the time, you should use one of those methods if you can.

VMCF
waits for a VMCF SENDX transmission. Any form of data transfer other than SENDX is rejected. The data received is typed on the console, and is also stacked in one of these forms (noting how the message was sent):

```
*VMCF userid message-text
```

or

```
*SMSG userid message-text
```

Control is then returned to CMS (or to a calling exec) with the return code set to 1.

When this option is given, WAKEUP issues a VMCF AUTHORIZE for any user ID. The maximum data that WAKEUP can receive through SENDX is 280 bytes.

Because the SMSG option implies VMCF, you do not need to specify the VMCF option if you specify SMSG.

Note: WAKEUP expects to be the only application waiting for VMCF communications in the virtual machine. If another application also enables for VMCF communications, unpredictable results may occur.

Items Stacked by WAKEUP: Seven WAKEUP options cause data to be stacked: EXT, FILE, IO, IUCVMSG, SMSG, TIME, and VMCF. WAKEUP stacks the data in the following order regardless of the order the options are specified when you invoke WAKEUP:

1. Current date and time
2. Line from the WAKEUP TIMES file, or an asterisk (*) if no line is found
3. An IUCV, SMSG or VMCF message, or EXT or IO interrupt data.

Note: In general, the last line stacked by WAKEUP is the one you really want to use, not the first line.

The WAKEUP TIMES File: Entries in a WAKEUP TIMES file are coded as follows:

Table 57. Format of Records in a WAKEUP TIMES File

1	10	19	28 (when stacked)
ALL	HH:MM:SS	datestamp	user-text (once a day)
MM/DD/YY	HH:MM:SS	datestamp	user-text (once)
==/DD/YY	HH:MM:SS	datestamp	user-text (once a month)
==/==/==	HH:MM:SS	datestamp	user-text (once a day)
==/01/==	HH:MM:SS	datestamp	user-text (on the 1st)
dayofweek	HH:MM:SS	datestamp	user-text (once a week)
WEEKEND	HH:MM:SS	datestamp	user-text (on weekends)
S-S	HH:MM:SS	datestamp	user-text (same as above)
WEEKDAY	HH:MM:SS	datestamp	user-text (on weekdays)
M-F	HH:MM:SS	datestamp	user-text (same as above)
==/==/==	+05	timestamp	user-text (every 5 minutes)
WEEKEND	+10:30	timestamp	user-text (" 10 mins 30 sec weekends)
WEEKDAY	+20	timestamp	user-text (" 20 mins weekdays)
dayofweek	+5	timestamp	user-text (" 5 mins on day)
M-F	+02:30:0	timestamp	user-text (" 150 mins on weekdays)

Each record in the WAKEUP TIMES file follows the format:

```
date time stamp rest-of-record
```

WAKEUP only looks at the `date`, `time`, and `stamp` fields. It uses these fields to determine if it should execute the record today, the time it should execute the record, and when it last executed the record.

Note: If the date and time fields indicate a time before the current time, the virtual machine will wake up immediately.

The Date Field (Columns 1–8): The date field is eight characters long and begins in column 1. It tells WAKEUP the date or day of the week when the record should be considered.

Use these valid formats for the date field:

MM/DD/YY
to specify the exact date. You can also use an equal sign (=) for any of the numbers to specify general dates. For example, specify ==/10/== to process something on the tenth of every month.

Note: If both the date and time fields are exact (for example, `03/15/87 03:45:30`), WAKEUP will change the first slash to a period (for example, `03.15/87`) when it processes the record. This makes it easy to write an exec to delete records that will never be executed again.

ALL
to specify every day. You can also use ==/==/== to specify every day.

Day Name
to specify a day of the week. Specify `MONDAY`, `TUESDAY`, `WEDNESDA` (you have to leave off the "Y"), `THURSDAY`, `FRIDAY`, `SATURDAY`, or `SUNDAY`. You can abbreviate any name to three letters.

WEEKEND or S-S
to specify Saturday and Sunday.

WEEKDAY or M-F
to specify every weekday.

MONTHLY
to specify once a month. If your system is always up on the first of the month, you can use ==/01/== instead of MONTHLY. MONTHLY is designed to ensure that the record will be processed once a month, even if your system happens to be down on the first of the month.

You must use the HH:MM:SS format in the time field with MONTHLY. (You cannot use relative time intervals.)

YEARLY
to specify once a year. If your system is always up on New Years Day, you can use 01/01/== instead of YEARLY.

You must use the HH:MM:SS format in the time field with YEARLY. (You cannot use relative time intervals.)

*****
to specify that this record is a comment. Comment records (those beginning with an asterisk) can be anywhere in a WAKEUP TIMES file.

The Time Field (Columns 10–17): The time field is also eight characters long and begins in column 10. It tells WAKEUP the time you want the record stacked.

Use these valid formats for the time field:

HH:MM:SS
to specify the exact time. For example, 13:05:00 will wake up at 1:05 p.m.

+MM
to specify every MM minutes. For example, +05 means do this every 5 minutes.

+HH:MM:S
to specify every HH hours, MM minutes, and S seconds. (The seconds can only be specified in multiples of 10. For example, when S is specified as 5, this would be equivalent to 50 seconds.)

Date/Time Stamp Field (Columns 19–26): The date or time stamp field is eight characters long and begins in column 19. WAKEUP records the last WAKEUP date or time here. If the time field contains an exact time (for example, 23:55:00), WAKEUP records a *date* stamp. If the time field contains a relative time (for example, +15), WAKEUP records a *time* stamp.

Note: This field should not be altered by the user to set the WAKEUP events. This should be done by coding the appropriate date field and time field entries.

The Rest of the Record (Columns 28–255): Information or commands describing the event can start in column 28 and can extend to column 255. Your application can put its own data here.

Note:
WAKEUP will only read the WAKEUP TIMES file for the current day. Therefore, if you are using the file option, an event must occur that will cause WAKEUP to be invoked on the next day and read the file. The sample exec in Figure 68 on page 1206 will cause the ID running WAKEUP to sleep through midnight, wake up the next day, re-execute the loop in the exec, and reinvoke WAKEUP on the new day, causing WAKEUP to read the WAKEUP TIMES file for the new day.

Messages and Return Codes

- DMS002E File *fn* not found [RC=28]
- DMS036E Open error code *nn* on *ddname*; FSOPEN return code = *nn* [RC=2*xx*.]
- DMS104S Error *nn* reading file *fn ft fm* [RC=207]
- DMS105S Error *nn* writing file *fn ft fm* [RC=211]
- DMS109S Insufficient free storage available [RC=4*xxx*]
- DMS205W Reader empty, reader not ready or empty reader file [RC=201]

- DMS394E Invalid option: *option* [RC=200]
- DMS514E Return code *nn* from *command* [RC=1*xxx*]
- DMS618E NUCEXT failed [RC=5*xxx*]
- DMS2186E Error *nn* from CP SPOOL RDR HOLD [RC=202]
- DMS2189E DMSRLD failed with return code *rc* [RC=1|RC=6*xxx*]
- DMS2246I *hh:mm:ss* WAKEUP {in|at} *hh:mm:ss* (*num* sec).
- DMS2247W The APPC option was entered without the EXT option for WAKEUP. The APPC option will be ignored
- DMS2248E Invalid {hours|minutes|seconds|format} in time parameter *parm*. [RC=203|204|205|206]
- DMS2249E Error *rc* from VMCF authorize. [RC=208]
- DMS2250E SENDX data length > 100 chars. [RC=209]
- DMS2251E VMCF data transfer error. [RC=210]
- DMS2252I Invalid time *hh:mm:ss* in *fileid*, record *num* (ignored). [RC=203|204|205|206]
- DMS2253E File *fileid* must be on a R/W disk. [RC=213]
- DMS2254E PSW error - Re-IPL CMS and restart. [RC=214]
- DMS2255E +MM form invalid with MONTHLY/YEARLY.
- DMS2256E Buffer already declared. [RC=300]
- DMS2257E IUCV Connect RC = *rc*. [RC=3*xx*]
- DMS2258E SET MSG IUCV failed RC = *rc*.
- DMS2259E Error *nn* during IUCV receive.
- DMS2326S Unexpected return code *rc* from {HNDIUCV CLEAR|HNDIUCV SET}. [RC=*xx*]

Return codes:

RC
Meaning

-2
First time invocation from CMS subset (WAKEUP cannot be invoked from CMS subset)

0
Nothing to wait for—all timer file options are done

1
VMCF/SMSG received and stacked

2
The specified time period has elapsed

3
A time from the timer file has been reached

4
A reader file of the proper class has arrived

5
A message has arrived

6
Interrupt from the virtual console

7
I/O interrupt

8
External interrupt

28
WAKEUP TIMES file not found

100
Explanation complete (when '?' specified)

101
Empty file in card reader, RDR option (RC from FSREAD)

103
Unknown error from card reader, RDR option (RC from FSREAD)

200
Invalid command option

201
Reader not operational

202
Error from CP SPOOL RDR HOLD

203
Invalid hours field of time parameter

204
Invalid minutes field of time parameter

205
Invalid seconds field of time parameter

206
Invalid time parameter

207
Error reading times file

208
VMCF AUTHORIZE error

209
VMCF SENDX error

210
VMCF data transfer error

211
Error writing times file

212
Invalid time parameter in times file

213
Times file not on R/W disk

214
PSW error

2xx
Return code xx from FSOPEN

300
IUCV Buffer error

3xx
Return code xx from IUCV operation

1xxx
CMSIUCV/HNDIUCV error occurred. 'xxx' is the IPRCODE field returned by IUCV to aid in error diagnosis or, return code xxx from FSSTATE

4xxx
Return code xxx from DMSRLD

4032
Storage not available for header record buffer

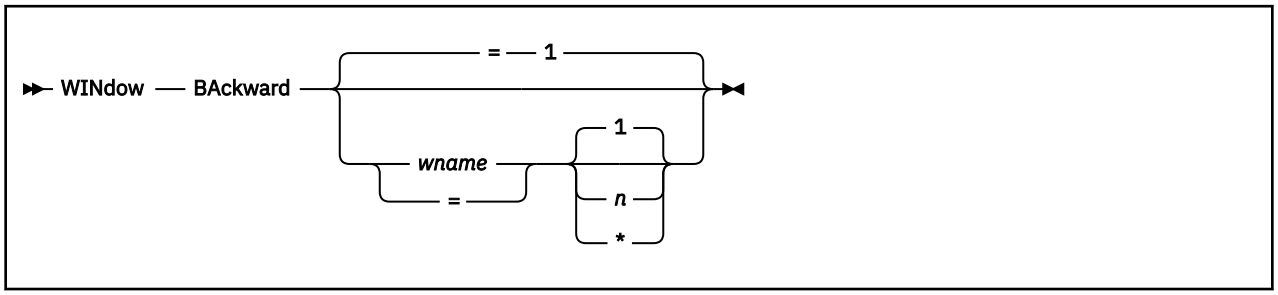
4033
Storage not available for module

4034
Storage not available for RLD buffer

5xxx
Return code xxx from NUCEXT SET or NUCEXT query

6xxx
Return code xxx from DMSRLD

WINDOW BACKWARD



Authorization

General User

Purpose

Use the WINDOW BACKWARD command to move the window toward the top of the virtual screen so the first data line displayed in the window becomes the last data line displayed.

Operands

wname
is the name of the window being scrolled. An = indicates the topmost window is scrolled. If this operand is not specified, = is the default. Only a window defined with the TOP option can qualify as the topmost window.

n
if you specify a number for *n*, the window is moved backward n displays. The * means scroll to the top. The default is 1 display.

Usage Notes

1. The WINDOW BACKWARD command is valid only when the window is connected to a virtual screen. For more information, see "WINDOW HIDE" on page 1230 and "WINDOW SHOW" on page 1245.
2. When you scroll a window backward, only the data area of the screen moves; the reserved lines (PF key definition area and title line) are static.
3. If the window has been cleared and you scroll the window using WINDOW BACKWARD, you scroll to the bottom of the virtual screen. For more information, see "WINDOW CLEAR" on page 1219.
4. If the window is positioned in the middle of the virtual screen and you scroll backward by specifying an "n" that would result in moving past the virtual screen 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.

Messages and Return Codes

- DMS386E Missing operand(s) [RC=24]
- DMS388E Invalid keyword: *keyword* [RC=24]
- DMS389E Invalid operand: *operand* [RC=24]
- DMS391E Unexpected operand(s): *operand* [RC=24]
- DMS921E Window *wname* is not defined [RC=28]
- DMS929E 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 their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

Return codes:

RC
Meaning

0
Normal

1
Virtual screen top or bottom reached, or window is cleared

WINDOW BOTTOM

Authorization

General User

Purpose

Use the WINDOW BOTTOM command to move 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.

Operands

wname
is the name of the window being scrolled. An = indicates the topmost window is scrolled. If this operand is not specified, = is the default. Only a window defined with the TOP option can qualify as the topmost window.

Usage Notes

1. The WINDOW BOTTOM command is valid only when the window is connected to a virtual screen. For more information, see "WINDOW HIDE" on page 1230 and "WINDOW SHOW" on page 1245.
2. If the window has been cleared and you scroll the window using WINDOW BOTTOM you scroll to the bottom of the virtual screen. For more information, see "WINDOW CLEAR" on page 1219.

Messages and Return Codes

- DMS386E Missing operand(s) [RC=24]
- DMS388E Invalid keyword: *keyword* [RC=24]
- DMS389E Invalid operand: *operand* [RC=24]
- DMS391E Unexpected operand(s): *operand* [RC=24]
- DMS921E Window *wname* is not defined [RC=28]
- DMS929E 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 their location are:

Reason	Location
Errors in command syntax	"Command Syntax Error Messages" on page 1411

Return codes:

RC
Meaning

0
Normal

1
Vscreen top or bottom reached, or window is cleared

WINDOW CLEAR

Authorization

General User

Purpose

Use the WINDOW CLEAR command to scroll past all data in the virtual screen to which the window is connected so no scrollable data is displayed in the window.

Operands

wname
is the name of the window that is cleared. An = indicates the topmost window is cleared. If this operand is not specified, = is the default. Only a window defined with the TOP option can qualify as the topmost window.

Usage Notes

1. WINDOW CLEAR is valid only when the window is connected to a virtual screen (see WINDOW HIDE and WINDOW SHOW).
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 WINDOW CLEAR 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. For more information, see "VSCREEN WRITE" on page 1192. The WINDOW CLEAR 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 an XEDIT session, you cannot clear the window XEDIT is using.

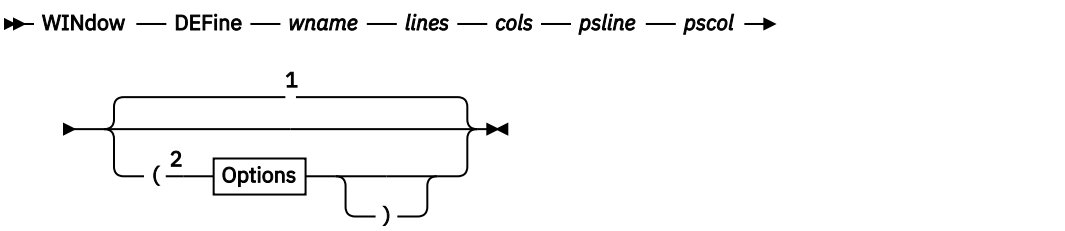
Messages and Return Codes

- DMS386E Missing operand(s) [RC=24]
- DMS388E Invalid keyword: *keyword* [RC=24]
- DMS391E Unexpected operand(s): *operand* [RC=24]
- DMS921E Window *wname* is not defined [RC=28]
- DMS929E Window *wname* is not connected to a virtual screen [RC=36]

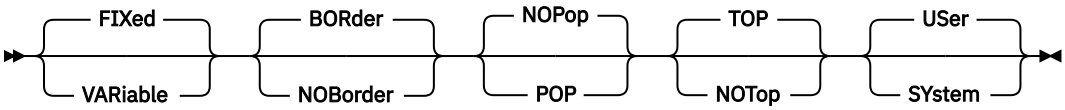
Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	"Command Syntax Error Messages" on page 1411

WINDOW DEFINE



Options



Notes:

[1] The defaults you receive appear above the line in the Options fragment.
[2] You can enter Options in any order between the parentheses.

Authorization

General User

Purpose

Use the WINDOW DEFINE command to create a window with the specified name, size, and position on the physical screen.

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” on page 1221.

pscol
is the column on the physical screen where the left edge of the window is positioned.

Options

VARiable
indicates the number of lines in the window may vary depending on the amount of scrollable data displayed.

FIXed
indicates the number of lines in the window is always constant. The default is FIXED.

BORder
indicates the borders are displayed when possible. The default is BORDER.

NOBorder
indicates the borders are not displayed.

POP
specifies the window is displayed on top of all other windows when the virtual screen the window is showing is updated.

NOPop
specifies there is no effect on the window's position in the ordered list of windows when the virtual screen the window is showing is updated. The default is NOPOP.

TOP
specifies the window may qualify as the topmost window. This is the default. Most windowing commands process the topmost window by default or when = is specified as the window name. For more information, see Usage Note “13” on page 1222.

NOTop
specifies 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 the window is deleted when a task abnormally ends (abend) or when the HX (halt execution) command is issued. This is the default.

SYstem
indicates 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 * or = is not valid.
3. Defining a window does not automatically display it. To display a window, it must be connected to a virtual screen. For more information, see “WINDOW HIDE” on page 1230 and “WINDOW SHOW” on page 1245.
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. For more information on how to change the way in which reserved lines are displayed in a window, see “SET RESERVED” on page 985.
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. The variable *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 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. The borders are highlighted and displayed using these 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. For more information on the border commands, see "Window Border Commands" on page 1251 and *z/VM: CMS User's Guide*.
10. If the window, virtual screen, and physical screen do not have the same number of columns, it is recommended you define the window with one column greater than the number of columns in the virtual screen 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 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.
13. TOP and NOTOP do not have an effect on the order of displayed windows. For example, a window defined with the NOTOP option can be located on top of all other displayed windows.

Messages and Return Codes

- DMS014E Invalid function *function* [RC=24]
- DMS386E Missing operand(s) [RC=24]
- DMS389E Invalid operand: *operand* [RC=24]
- DMS391E Unexpected operand(s): *operand* [RC=24]
- DMS394E Invalid option: *option* [RC=24]
- DMS622E Insufficient free storage [RC=104]
- DMS676E Invalid character {*|=} for window name [RC=20]
- DMS915E Maximum number of windows already defined [RC=13]
- DMS920E Window *wname* already exists [RC=3]
- DMS922E Window does not fit entirely on the screen [RC=32]
- DMS926E 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 their location are:

Reason	Location
Errors in command syntax	"Command Syntax Error Messages" on page 1411

WINDOW DELETE

Authorization

General User

Purpose

Use the WINDOW DELETE command to remove a window definition.

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).

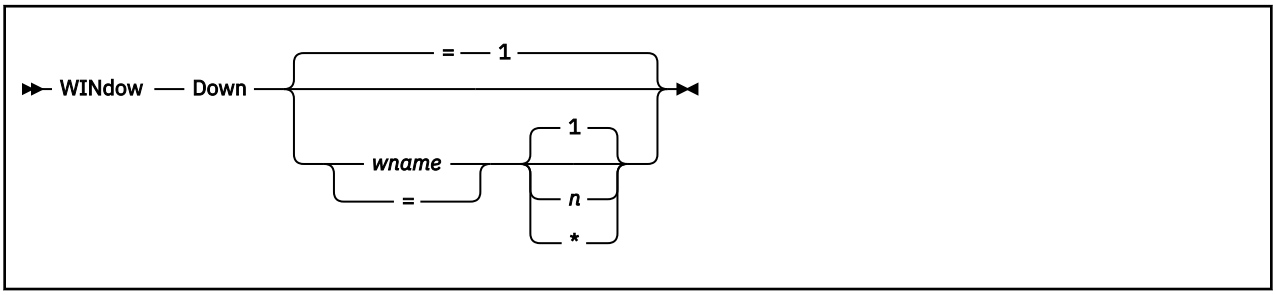
Messages and Return Codes

- DMS386E Missing operand(s) [RC=24]
- DMS388E Invalid keyword: *keyword* [RC=24]
- DMS391E Unexpected operand(s): *operand* [RC=24]
- DMS919E The CMS window cannot be deleted [RC=24]
- DMS921E Window *wname* is not defined [RC=28]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

WINDOW DOWN



Authorization

General User

Purpose

Use the WINDOW DOWN command to move the window toward the bottom of the virtual screen the number of data lines specified.

Operands

wname
is the name of the window being scrolled. An = indicates the topmost window is scrolled. If this operand is not specified, = is the default. Only a window defined with the TOP option can qualify as the topmost window.

n
if you specify a number for *n*, the window is scrolled down "n" lines. The * means scroll to the bottom. The default is 1 line.

Usage Notes

1. WINDOW DOWN is the same as WINDOW NEXT.
2. The WINDOW DOWN command is valid only when the window is connected to a virtual screen. For more information, see “WINDOW HIDE” on page 1230 and “WINDOW SHOW” on page 1245.
3. When you scroll forward, only the data area of the screen moves; the reserved lines (PF key definition area and title line) are static.
4. 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.
5. When you receive the status area message "Scroll forward for more information in vscreen *vname*" (in full-screen CMS) and there are multiple windows showing the specified virtual screen, it is recommended you scroll forward the window closest to the top of the ordered list of windows. This enables data in the virtual screen queue to be written to the virtual screen and displayed.
6. If the window has been cleared WINDOW DOWN has no effect. For more information, see “WINDOW CLEAR” on page 1219.

Messages and Return Codes

- DMS386E Missing operand(s) [RC=24]
- DMS388E Invalid keyword: *keyword* [RC=24]
- DMS389E Invalid operand: *operand* [RC=24]

- DMS391E Unexpected operand(s): *operand* [RC=24]
- DMS921E Window *wname* is not defined [RC=28]
- DMS929E 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 their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

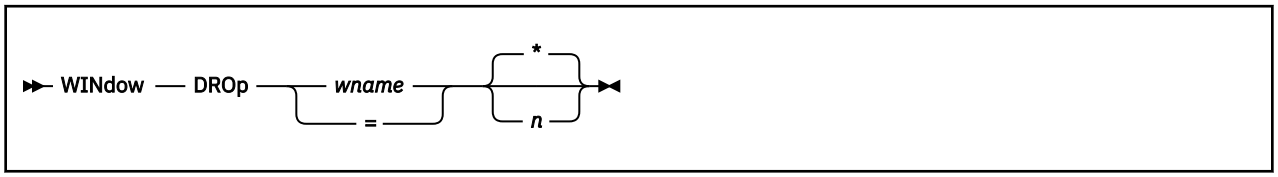
Return codes:

RC
Meaning

0
Normal

1
Vscreen top or bottom reached, or window is cleared

WINDOW DROP



Authorization

General User

Purpose

Use the WINDOW DROP command to move a window down in the order of displayed windows.

Operands

wname
is the name of the window to be dropped.

=
indicates the topmost window is dropped. Only a window defined with the TOP option can qualify as the topmost window.

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 the default.

Usage Notes

1. Dropping the WM window hides the WM window so 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 WINDOW POP WM. The *n* and * are ignored when specified with WM.
2. If the window is hidden, WINDOW DROP has no effect. For more information, see "WINDOW HIDE" on page 1230.
3. 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.
4. 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.

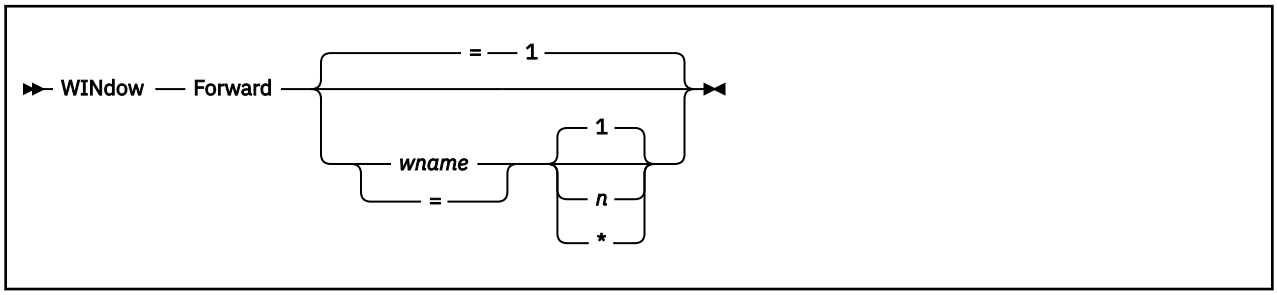
Messages and Return Codes

- DMS386E Missing operand(s) [RC=24]
- DMS388E Invalid keyword: *keyword* [RC=24]
- DMS389E Invalid operand: *operand* [RC=24]
- DMS391E Unexpected operand(s): *operand* [RC=24]
- DMS921E Window *wname* is not defined [RC=28]
- DMS929E 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 their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

WINDOW FORWARD



Authorization

General User

Purpose

Use the WINDOW FORWARD command to move the window toward the bottom of the virtual screen so the last data line displayed in the window becomes the first data line displayed.

Operands

wname
is the name of the window being scrolled. An = indicates the topmost window is scrolled. If this operand is not specified, = is the default. Only a window defined with the TOP option can qualify as the topmost window.

n
if you specify a number for *n*, the window is scrolled forward "n" displays. The default is 1. The * means scroll to the bottom.

Usage Notes

1. The WINDOW FORWARD command is valid only when the window is connected to a virtual screen. For more information, see “WINDOW HIDE” on page 1230 and “WINDOW SHOW” on page 1245.
2. When you scroll forward, only the data area of the screen moves; the reserved lines (PF key definition area and title line) are static.
3. If the window has been cleared and you scroll the window using WINDOW FORWARD, you scroll to the top of the virtual screen. For more information, see “WINDOW CLEAR” on page 1219.
4. If the window is positioned in the middle of the virtual screen and you scroll forward by specifying an "n" that would result 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. For more information, see “WINDOW CLEAR” on page 1219.
5. When you receive the status area message "Scroll forward for more information in vscreen *vname*" (in full-screen CMS) and there are multiple windows showing the specified virtual screen, it is recommended you scroll forward the window closest to the top of the ordered list of windows. This enables data in the virtual screen queue to be written to the virtual screen and displayed.

Messages and Return Codes

- DMS386E Missing operand(s) [RC=24]

- DMS388E Invalid keyword: *keyword* [RC=24]
- DMS389E Invalid operand: *operand* [RC=24]
- DMS391E Unexpected operand(s): *operand* [RC=24]
- DMS921E Window *wname* is not defined [RC=28]
- DMS929E 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 their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

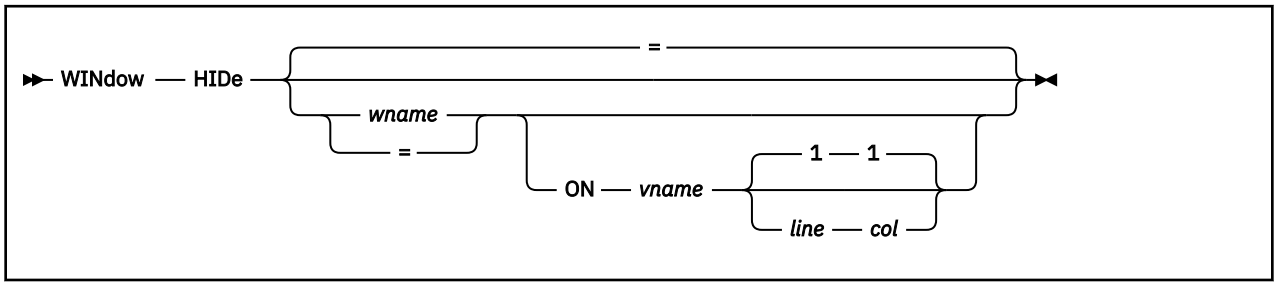
Return codes:

RC
Meaning

0
Normal

1
Vscreen top or bottom reached, or window is cleared

WINDOW HIDE



Authorization

General User

Purpose

Use the WINDOW HIDE command to prevent the specified window from being displayed and to connect the window to a virtual screen. The WINDOW SHOW command will redisplay the window.

Operands

wname
is the name of the window to be hidden. An = indicates the topmost window is hidden. If *wname* is not specified, = is the default. Only a window defined with the TOP option can qualify as the topmost window.

ON *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 virtual screen column number 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 WINDOW HIDE 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 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 WINDOW POP or WINDOW DROP 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 WINDOW POP STATUS command.

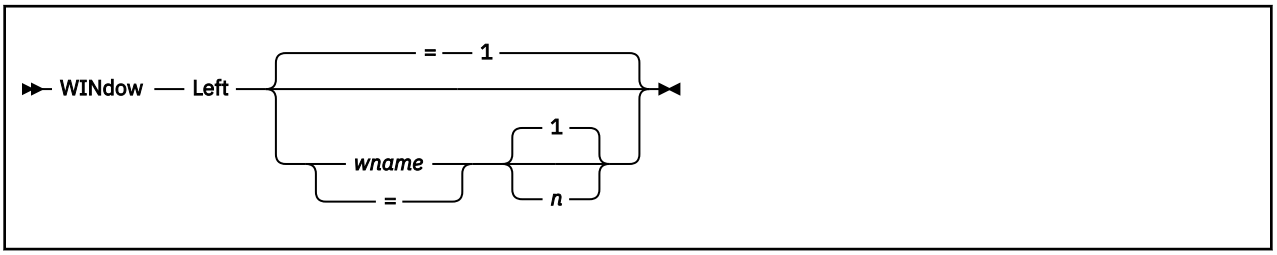
Messages and Return Codes

- DMS386E Missing operand(s) [RC=24]
- DMS388E Invalid keyword: *keyword* [RC=24]
- DMS389E Invalid operand: *operand* [RC=24]
- DMS391E Unexpected operand(s): *operand* [RC=24]
- DMS921E Window *wname* is not defined [RC=28]
- DMS921E Virtual screen *vname* is not defined [RC=28]
- DMS921E Window *wname* is not defined [RC=28]
- DMS923E Specified location is outside the virtual screen [RC=32]
- DMS929E 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 their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

WINDOW LEFT



Authorization

General User

Purpose

Use the WINDOW LEFT command to move the window toward the left edge of the virtual screen the number of columns specified.

Operands

wname
is the name of the window being scrolled. An = indicates the topmost window is scrolled. If this operand is not specified, = is the default. Only a window defined with the TOP option can qualify as the topmost window.

n
is the number of columns to be scrolled. The default for *n* is 1. If *n* is specified as 0, the window is moved to column 1.

Usage Notes

1. The WINDOW LEFT command is valid only when the window is connected to a virtual screen. For more information, see “WINDOW HIDE” on page 1230 and “WINDOW SHOW” on page 1245.
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.

Messages and Return Codes

- DMS386E Missing operand(s) [RC=24]
- DMS388E Invalid keyword: *keyword* [RC=24]
- DMS389E Invalid operand: *operand* [RC=24]
- DMS391E Unexpected operand(s): *operand* [RC=24]
- DMS921E Window *wname* is not defined [RC=28]
- DMS929E 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 their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

Return codes:

RC	Meaning
0	Normal
3	Scroll amount truncated

WINDOW MAXIMIZE

Authorization

General User

Purpose

Use the WINDOW MAXIMIZE command to expand a window to the physical screen size.

Operands

wname
is the name of the window to be expanded. An = indicates the topmost window is expanded. If this operand is not specified, = is the default. Only a window defined with the TOP option can qualify as the topmost window.

Usage Notes

1. The WINDOW RESTORE command returns the window to its size and location before the maximize.
2. A window can be maximized whether it is connected to a virtual screen and whether 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.
   - It contains only three data lines.
5. When you maximize a window that is not showing the active virtual screen so it covers the window or windows showing the active virtual screen, the WM window is automatically displayed. For more information, see Things You Should Know About Full-Screen CMS.

Messages and Return Codes

- DMS386E Missing operand(s) [RC=24]
- DMS388E Invalid keyword: *keyword* [RC=24]
- DMS391E Unexpected operand(s): *operand* [RC=24]
- DMS921E Window *wname* is not defined [RC=28]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

WINDOW MINIMIZE

Authorization

General User

Purpose

Use the WINDOW MINIMIZE command to reduce the size of the window to one line.

Operands

wname
is the name of the window to be reduced. An = indicates the topmost window will be reduced. If this operand is not specified, = is the default. Only a window defined with the TOP option can qualify as the topmost window.

Usage Notes

The WINDOW RESTORE command returns the window to its size and location before issuing the MINIMIZE command.

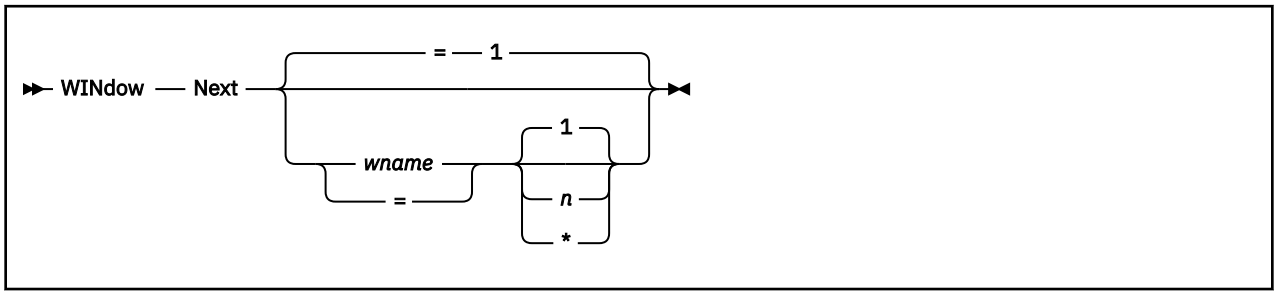
Messages and Return Codes

- DMS386E Missing operand(s) [RC=24]
- DMS388E Invalid keyword: *keyword* [RC=24]
- DMS391E Unexpected operand(s): *operand* [RC=24]
- DMS921E Window *wname* is not defined [RC=28]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

WINDOW NEXT



Authorization

General User

Purpose

Use the WINDOW NEXT command to move the window toward the bottom of the virtual screen the number of lines specified.

Operands

wname
is the name of the window being scrolled. An = indicates the topmost window is scrolled. If this operand is not specified, = is the default. Only a window defined with the TOP option can qualify as the topmost window.

n
if you specify a number for *n*, the window is scrolled down "n" lines. The * means scroll to the bottom. The default is 1 line.

Usage Notes

1. WINDOW NEXT is the same as WINDOW DOWN.
2. The WINDOW NEXT command is valid only when the window is connected to a virtual screen. For more information, see “WINDOW HIDE” on page 1230 and “WINDOW SHOW” on page 1245.
3. With WINDOW NEXT, only the data area of the screen moves; the reserved lines (PF key definition area and title line) are static.
4. If the window is positioned in the middle of the virtual screen and you scroll forward by specifying an "n" which results in scrolling past the bottom, the window is repositioned at the virtual screen bottom, and stops.
5. When you receive the status area message "Scroll forward for more information in vscreen *vname*" (in full-screen CMS) and there are multiple windows showing the specified virtual screen, it is recommended you scroll forward the window closest to the top of the ordered list of windows. This enables data in the virtual screen queue to be written to the virtual screen and displayed.
6. If the window has been cleared, WINDOW NEXT has no effect. For more information, see “WINDOW CLEAR” on page 1219.

Messages and Return Codes

- DMS386E Missing operand(s) [RC=24]
- DMS388E Invalid keyword: *keyword* [RC=24]
- DMS389E Invalid operand: *operand* [RC=24]

- DMS391E Unexpected operand(s): *operand* [RC=24]
- DMS921E Window *wname* is not defined [RC=28]
- DMS929E 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 their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

Return codes:

RC
Meaning

0
Normal

1
Vscreen top or bottom reached, or window is cleared

WINDOW POP

Authorization

General User

Purpose

Use the WINDOW POP command to move a window up in the order of displayed windows.

Operands

wname
is the name of the window to be moved up.

Popping the WM window allows you to enter commands from the command line or with WMPF keys. The *n* and * have no effect when specified with WM. For a list of commands you can enter in the WM environment, see Usage Note "5" on page 1239.

n
is the number of positions the window is to be moved up. An * indicates the window will be positioned on top of the other displayed windows. If this operand is not specified, * is the default.

Usage Notes

1. If the window is hidden, WINDOW POP has no effect. For more information, see "WINDOW HIDE" on page 1230.
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 WINDOW POP 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 WINDOW CLEAR command or by scrolling forward), the WINDOW POP command moves the window up in the order of displayed windows and scrolls it to the bottom of the virtual screen it is showing.
4. When you are using full-screen CMS and you enter the WINDOW POP 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 you specified may not be displayed at the top of the display order because another has been popped afterward.
5. You can display the WM window whether 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:

CP	WINDOW CLEAR
HELP	WINDOW DOWN
PSCREEN PUT	WINDOW DROP
QUERY BORDER	WINDOW FORWARD
QUERY HIDE	WINDOW HIDE

QUERY LOCATION	WINDOW LEFT
QUERY RESERVED	WINDOW MAXIMIZE
QUERY SHOW	WINDOW MINIMIZE
QUERY WINDOW	WINDOW NEXT
QUERY WMPF	WINDOW POP
SET BORDER	WINDOW POSITION
SET LOCATION	WINDOW RESTORE
SET RESERVED	WINDOW RIGHT
SET WINDOW	WINDOW SHOW
SET WMPF	WINDOW SIZE
WINDOW BACKWARD	WINDOW TOP
WINDOW BOTTOM	WINDOW UP

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 environment, issue the command WINDOW DROP WM or press the PF3 key (if the setting of WMPF key 3 has not been changed from the initial value). You may also press the PA2 or CLEAR key to scroll the topmost window forward; when the bottom of the window is reached, you will exit the WM environment.

6. The WINDOW POP WM command automatically defines the WM window and WM virtual screen if they do not already exist.
7. When you pop a window that is not showing the active virtual screen so it covers the window or windows showing the active virtual screen, the WM window is automatically displayed. For more information, see Things You Should Know About Full-Screen CMS.

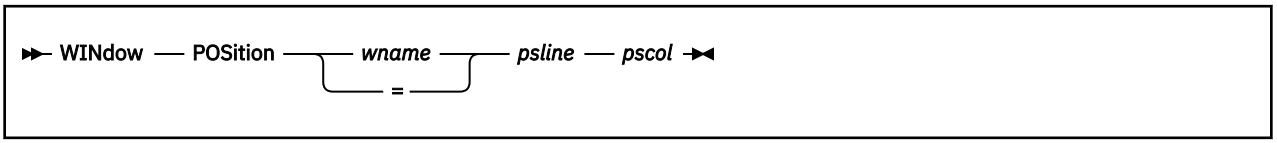
Messages and Return Codes

- DMS386E Missing operand(s) [RC=24]
- DMS388E Invalid keyword: *keyword* [RC=24]
- DMS389E Invalid operand: *operand* [RC=24]
- DMS391E Unexpected operand(s): *operand* [RC=24]
- DMS921E Window *wname* is not defined [RC=28]
- DMS929E 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 their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

WINDOW POSITION



Authorization

General User

Purpose

Use the WINDOW POSITION command to change the location of a window on the physical screen.

Operands

wname
is the name of the window to be moved. An = indicates the topmost window is moved. Only a window defined with the TOP option can qualify as the topmost window.

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. The variable *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

- DMS386E Missing operand(s) [RC=24]
- DMS388E Invalid keyword: *keyword* [RC=24]
- DMS389E Invalid operand: *operand* [RC=24]
- DMS391E Unexpected operand(s): *operand* [RC=24]
- DMS921E Window *wname* is not defined [RC=28]
- DMS922E 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 their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

WINDOW RESTORE

Authorization

General User

Purpose

Use the WINDOW RESTORE command to return a maximized or minimized window to its size and location before the maximize or minimize.

Operands

wname
is the name of the window to be restored. An = indicates the topmost window will be restored. The = is the default. Only a window defined with the TOP option can qualify as the topmost window.

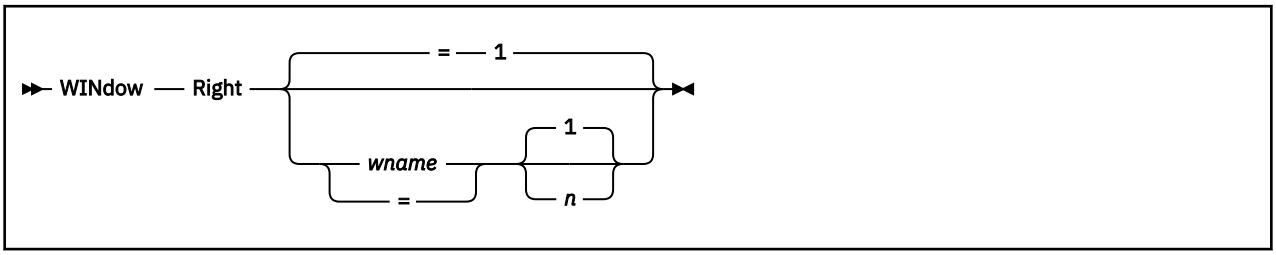
Messages and Return Codes

- DMS386E Missing operand(s) [RC=24]
- DMS388E Invalid keyword: *keyword* [RC=24]
- DMS391E Unexpected operand(s): *operand* [RC=24]
- DMS921E Window *wname* is not defined [RC=28]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

WINDOW RIGHT



Authorization

General User

Purpose

Use the WINDOW RIGHT command to move the window toward the right edge of the virtual screen the number of columns specified.

Operands

wname
is the name of the window being scrolled. An = indicates the topmost window is scrolled. If this operand is not specified, = is the default. Only a window defined with the TOP option can qualify as the topmost window.

n
is the number of columns to be scrolled. The default for *n* is 1. If *n* is specified as 0, the window is moved to column 1.

Usage Notes

1. The WINDOW RIGHT command is valid only when the window is connected to a virtual screen. For more information, see "WINDOW HIDE" on page 1230 and "WINDOW SHOW" on page 1245.
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.

Messages and Return Codes

- DMS386E Missing operand(s) [RC=24]
- DMS388E Invalid keyword: *keyword* [RC=24]
- DMS389E Invalid operand: *operand* [RC=24]
- DMS391E Unexpected operand(s): *operand* [RC=24]
- DMS921E Window *wname* is not defined [RC=28]
- DMS929E 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 their location are:

Reason	Location
Errors in command syntax	"Command Syntax Error Messages" on page 1411

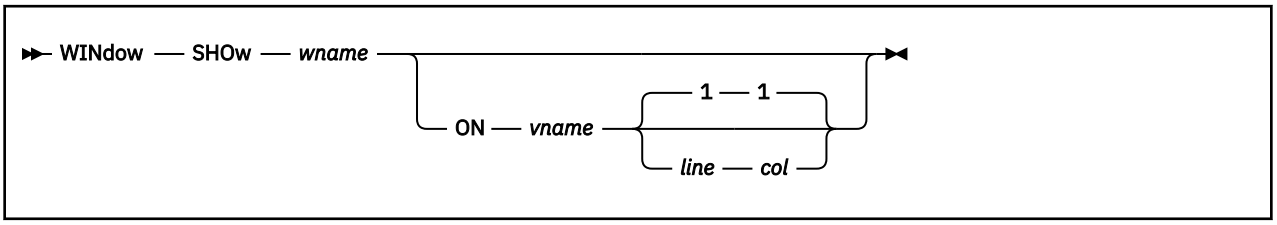
Return codes:

RC
Meaning

0
Normal

3
Scroll amount truncated

WINDOW SHOW



Authorization

General User

Purpose

Use the WINDOW SHOW command to place a window at the top of the window display order and to connect the window to a virtual screen.

Operands

wname
is the name of the window.

ON *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

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 WINDOW SHOW 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. Also, if there is no data on the virtual screen, the line defaults to 1. If the line specified is past the bottom line that has been written to the virtual screen, the window will be connected to that existing bottom line.
4. 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 WINDOW SHOW 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.
5. When you are using full-screen CMS and you enter the WINDOW SHOW 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 you specified may not be displayed at the top of the display order because another has been popped afterwards.

Messages and Return Codes

- DMS386E Missing operand(s) [RC=24]

- DMS388E Invalid keyword: *keyword* [RC=24]
- DMS389E Invalid operand: *operand* [RC=24]
- DMS391E Unexpected operand(s): *operand* [RC=24]
- DMS921E Window *wname* is not defined [RC=28]
- DMS921E Virtual screen *vname* is not defined [RC=28]
- DMS921E Window *wname* is not defined [RC=28]
- DMS923E Specified location is outside the virtual screen [RC=32]
- DMS929E 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 their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

WINDOW SIZE

Authorization

General User

Purpose

Use the WINDOW SIZE command to change the number of lines and columns for a specified window.

Operands

wname
is the name of the window. An = indicates the size of the topmost window is changed. Only a window defined with the TOP option can qualify as the topmost window.

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 Notes

The window's size and location must be such that, excluding borders, the entire window fits on the physical screen.

Messages and Return Codes

- DMS386E Missing operand(s) [RC=24]
- DMS388E Invalid keyword: *keyword* [RC=24]
- DMS389E Invalid operand: *operand* [RC=24]
- DMS391E Unexpected operand(s): *operand* [RC=24]
- DMS921E Window *wname* is not defined [RC=28]
- DMS922E 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 their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

WINDOW TOP

Authorization

General User

Purpose

Use the WINDOW TOP command to move 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.

Operands

wname
is the name of the window being scrolled. An = indicates the topmost window is scrolled. If this operand is not specified, = is the default. Only a window defined with the TOP option can qualify as the topmost window.

Usage Notes

1. The WINDOW TOP command is valid only when the window is connected to a virtual screen. For more information, see "WINDOW HIDE" on page 1230 and "WINDOW SHOW" on page 1245.
2. If the window has been cleared and you scroll the window using WINDOW TOP, you scroll to the top of the virtual screen. For more information, see "WINDOW CLEAR" on page 1219.

Messages and Return Codes

- DMS386E Missing operand(s) [RC=24]
- DMS388E Invalid keyword: *keyword* [RC=24]
- DMS389E Invalid operand: *operand* [RC=24]
- DMS391E Unexpected operand(s): *operand* [RC=24]
- DMS921E Window *wname* is not defined [RC=28]
- DMS929E 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 their location are:

Reason	Location
Errors in command syntax	"Command Syntax Error Messages" on page 1411

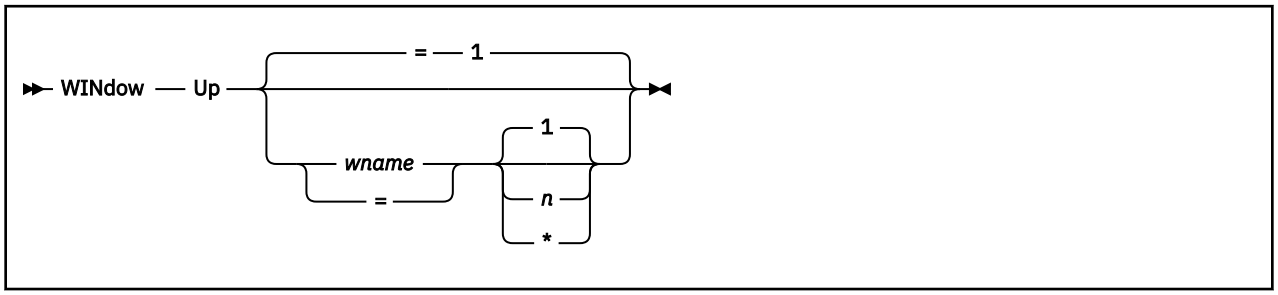
Return codes:

RC
Meaning

0
Normal

1
Vscreen top or bottom reached, or window is cleared

WINDOW UP



Authorization

General User

Purpose

Use the WINDOW UP command to move the window toward the top of the virtual screen the number of data lines specified.

Operands

wname
is the name of the window being scrolled. An = indicates the topmost window is scrolled. If this operand is not specified, = is the default. Only a window defined with the TOP option can qualify as the topmost window.

n
if you specify a number for *n*, the window is scrolled up "n" lines. The * means scroll to the top. The default is 1 line.

Usage Notes

1. The WINDOW UP command is valid only when the window is connected to a virtual screen. For more information, see “WINDOW HIDE” on page 1230 and “WINDOW SHOW” on page 1245.
2. When you scroll a window backward, only the data area of the screen moves; the reserved lines (PF key definition area and title line) are static.
3. If the window is positioned in the middle of the virtual screen and you scroll backward specifying an "n", which would result in moving past the virtual screen top, the window is repositioned at the virtual screen top and stops.
4. If the window has been cleared, WINDOW UP has no effect. For more information, see “WINDOW CLEAR” on page 1219.

Messages and Return Codes

- DMS386E Missing operand(s) [RC=24]
- DMS388E Invalid keyword: *keyword* [RC=24]
- DMS389E Invalid operand: *operand* [RC=24]
- DMS391E Unexpected operand(s): *operand* [RC=24]
- DMS921E Window *wname* is not defined [RC=28]
- DMS929E 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 their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

Return codes:

RC
Meaning

0
Normal

1
Vscreen top or bottom reached, or window is cleared

Window Border Commands

Purpose

Border commands are single-character commands you enter in the corners of the window border to control the window's movement and characteristics.

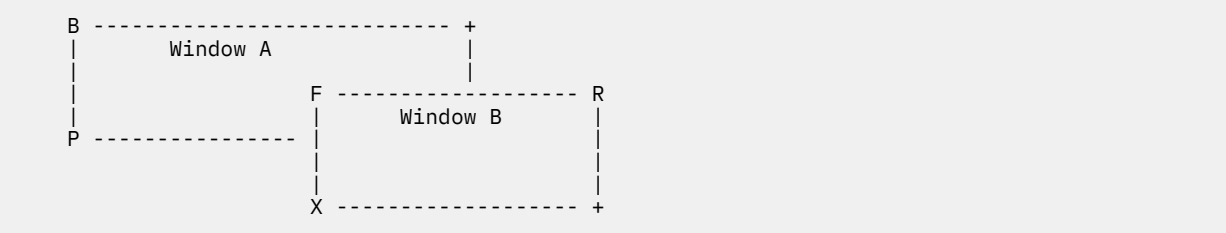
The commands are processed as soon as they are entered.

The border commands are:

Usage Notes

1. Use the SET BORDER command to control the characteristics of the borders 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 Enter.
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.

Messages and Return Codes

- DMS931E Invalid border command: *character*

For more information about and examples of how to use the window border commands, see *z/VM: CMS User's Guide*.

B

Authorization

General User

Purpose

Use the B border command to scroll the window backward.

Usage Notes

1. If the window has been cleared, the B command scrolls you to the bottom of the virtual screen. For more information, see "C" on page 1252.
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

Authorization

General User

Purpose

Use the C border command to clear the window of all scrollable data.

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 XEDIT session, you cannot clear the window XEDIT is using.

D

Authorization

General User

Purpose

Use the D border command to drop a window, or place it beneath all other windows.

Usage Notes

When using full-screen CMS, you cannot drop a window 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

Authorization

General User

Purpose

Use the F border command to scroll the window forward.

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. For more information, see “WINDOW CLEAR” on page 1219.

H

Authorization

General User

Purpose

Use the H border command to hide the window.

L

Authorization

General User

Purpose

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.

Usage Notes

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

Authorization

General User

Purpose

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.

Usage Notes

The window's size and location must be such that, excluding borders, the entire window fits on the physical screen.

Messages and Return Codes

- DMS922E Window does not fit entirely on the screen

N

Authorization

General User

Purpose

Use the N border command to minimize the window.

Usage Notes

The O command returns the window to its size and location before the minimize.

O

Authorization

General User

Purpose

Use the O border command to restore a maximized or minimized window to its original size and location.

P

Authorization

General User

Purpose

Use the P border command to pop a window, or place it on top of all other windows.

Usage Notes

When you pop a window that is not showing the active virtual screen so it covers the window or windows showing the active virtual screen, the WM window is automatically displayed. For more information, see Things You Should Know About Full-Screen CMS.

R

Authorization

General User

Purpose

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.

Usage Notes

1. 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

Authorization

General User

Purpose

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.

Usage Notes

The window's size and location must be such that, excluding borders, the entire window fits on the physical screen.

Messages and Return Codes

- DMS922E Window does not fit entirely on the screen
- DMS930E Cursor is not in a valid location

X

Authorization

General User

Purpose

Use the X border command to maximize a window.

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.

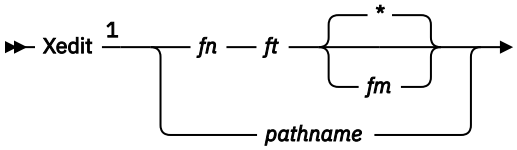
   For example, if a variable size window is connected to line 1 of a virtual screen which contains three data lines, when it is maximized:

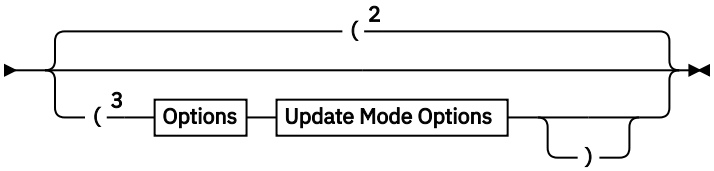
   - It moves to line 1, column 1 of the physical screen
   - Its width is the size of the physical screen

- It contains only three data lines

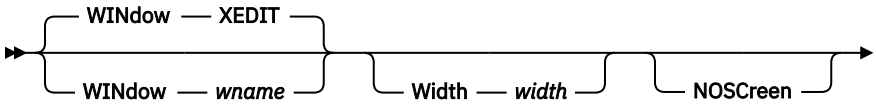
4. When you maximize a window that is not showing the active virtual screen so it covers the window or windows showing the active virtual screen, the WM window is automatically displayed. For more information, see Things You Should Know About Full-Screen CMS.

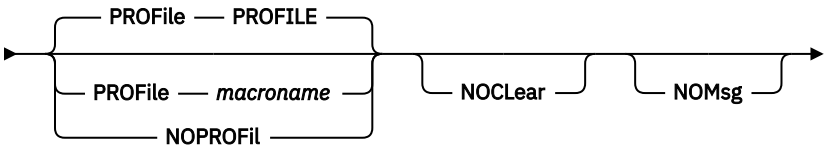
XEDIT

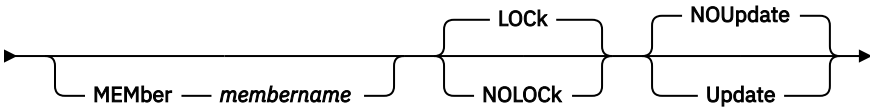


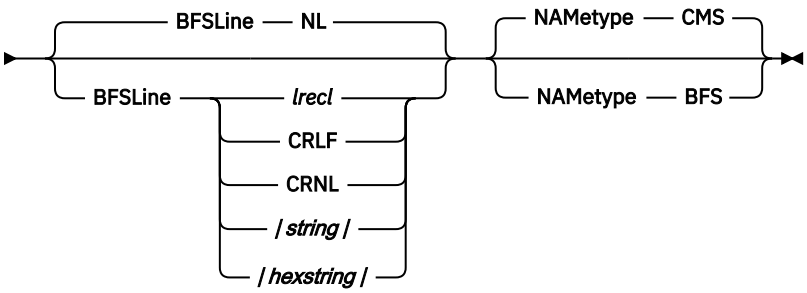


Options









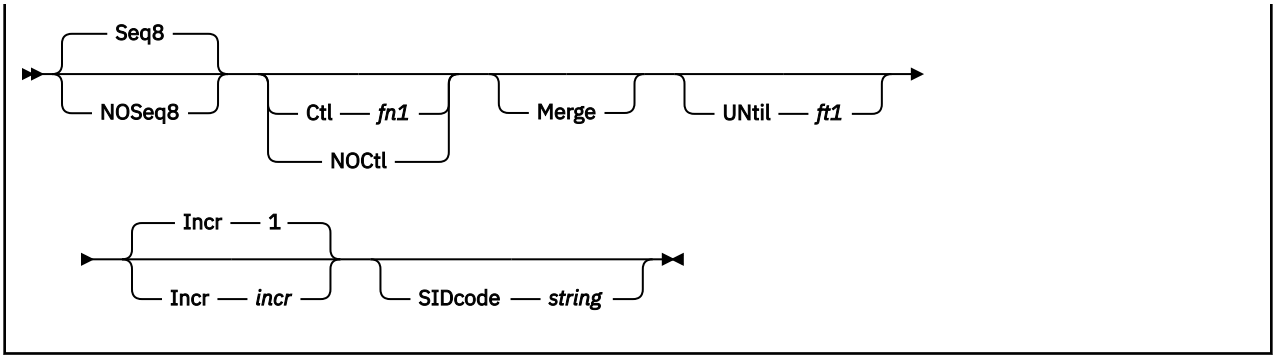
Notes:

[1] If a profile is invoked which issues a LOAD subcommand that specifies *fn* and *ft* or *pathname*, *fn* and *ft* or *pathname* are not required.

[2] The default options are shown above the main line in the options groups.

[3] You can enter options in any order between the parentheses. If a default is not shown for an option, refer to the option description for the information.

Update Mode Options



Purpose

Use XEDIT to invoke the editor to create, modify, and manipulate CMS files on disk, SFS files, or BFS regular files in the byte file system. Once you invoke the editor, you can execute XEDIT subcommands and use REXX or the EXEC 2 macro facility.

You can return control to the CMS environment by entering the XEDIT subcommand FILE, FFILE, QUIT, or QQUIT.

Operands

fn ft
are the file name and the file type of the SFS or minidisk 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 an accessed minidisk or SFS 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 disk or directory and its extensions are searched. If the file mode is not specified or is specified as an asterisk (*), all accessed disks 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.

pathname
is the name of the BFS file to be edited. See “Understanding Byte File System (BFS) Path Name Syntax” on page 9.

Options

WINdow *wname*
is the name of the virtual screen and window XEDIT uses to display all files being edited in the ring. By default, XEDIT uses the window and virtual screen named "XEDIT".

Note: The window name must not contain file name characters that are not valid. Also, *CMS* or *WM* cannot be used as the window name.

For more information on virtual screens and windows, see Appendix G in *z/VM: XEDIT Commands and Macros Reference*.

Width *width*
defines the amount of virtual storage containing 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. For SFS and minidisk files, if WIDTH is not specified in either the XEDIT command or the LOAD subcommand, the default is the larger of the following:

- Logical record length (LRECL) of the file
- Default logical record length associated with the file type. See Appendix A in *z/VM: XEDIT Commands and Macros Reference* for a list of these defaults.

For BFS files, if WIDTH is not specified in either the XEDIT command or the LOAD subcommand, the default depends upon the BFSLINE value.

- For BFSLINE *lrecl*, the default is *lrecl*
- Otherwise, the default value is 80 or the length of the longest line, whichever is larger

NOSCreen
forces a 3270 display terminal into line (typewriter) mode.

PROFile *macroname*
If the specified macro exists on one of the accessed minidisks or SFS directories, the editor executes it as the first subcommand. If the specified macro exists and is empty, an error message is displayed.

If the specified macro is not found on an accessed CMS disk or SFS 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. If the macro exists and is empty, the macro is ignored.

The profile macro must always have a file type of XEDIT. The profile macro may not reside in the byte file system.

NOPROFil
forces the editor not to execute the default PROFILE macro.

NOCLear
specifies 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. The option only affects an emulator session.

NOMsg
enters a file with a default of SET MSGMODE OFF.

MEMber *membername*
is the name of a member in the macro library specified in *fn ft fm*. If MEMBER is specified, *ft* must be MACLIB. When the MEMBER option is specified, XEDIT scans the specified MACLIB to find the member. If the member is found, XEDIT reads it into storage. If the member does not exist in that library, a new member is created. The member is displayed with a file ID of *membername* MEMBER *fm*.

CMS supports SFS MACLIBs which do not have members; however, CMS does not support empty members. If the MACLIB file is empty, an error message is displayed.

MEMBER is ignored if you attempt to use it when editing a BFS file.

LOCk
causes the editor to lock an existing SFS file to prevent other users from modifying the file while you are editing it, or causes the editor to obtain an advisory lock for an existing BFS file while it is being read into the XEDIT session. Note that a BFS file does not remain locked while in the session, and other users may choose to override an advisory lock.

The LOCK option is ignored for files on minidisks and files that reside in SFS directory control directories and files in NFS-mounted directories.

For files in SFS file control directories, 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.

For SFS files, 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 you do not want the editor to try to lock the file.

For SFS files:
You can use this option to edit a file another user has locked SHARE or UPDATE. If you specify NOLOCK, other users can change the file while you are editing it. NOLOCK is the default for files that reside in SFS directory control directories.

You should only use this option if you are not going to make any changes to the file, or if you will save your changes under a different file identifier. Otherwise, any changes you make will not include modifications other users make to the permanent copy of the file during your editing session.

For BFS files:
The editor does not obtain an advisory lock while reading the file into the XEDIT session.

NOUpdate
specifies the editor is to apply no update statements (even if UPDATE is specified in the LOAD subcommand in the profile).

Update
The editor searches all accessed minidisks and SFS 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 modification the user makes 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 with a file mode of A1 to contain modifications the user makes. If the original source file (base file) is empty, an error message is displayed. If an update file is empty, an informational message is displayed. For more information on the XEDIT UPDATE option, see *z/VM: CMS Application Development Guide*.

UPDATE is ignored if you attempt to use it when editing a BFS file.

BFSLine
Use the BFSLINE option to tell XEDIT how to translate a BFS byte stream into records when reading the file into virtual storage for editing and to set the initial setting of SET BFSLINE.

You can define an end of line character or characters for use in interpreting lines in a BFS file if you specify anything other than BFSLINE *lrecl*. When a file is read into an XEDIT session, everything up to the end of line character is interpreted as a line and presented as a 'record' in the XEDIT session. The end of line character is not displayed while XEDITing the BFS file. When a FILE, PUT, PUTD, or SAVE subcommand is entered for the file when it is being written back to a byte file system, the setting of SET BFSLINE is used to determine the end of line character that will be used. If the setting has not been changed from its initial value, the same end of line character that existed previously will be used. See the description for SET BFSLINE in *z/VM: XEDIT Commands and Macros Reference* for more information.

BFSLINE *lrecl* does not separate the file into records based on an end of line character.

The BFSLINE option determines the initial setting used for record format (RECFM). For BFSLINE *lrecl*, this initial setting is fixed (F). For any other BFSLINE value, it is variable (V). Changing this value while in an XEDIT session will affect the way the file is stored in the byte file system. See the SET RECFM command description in *z/VM: XEDIT Commands and Macros Reference* for more information.

If not specified, BFSLINE NL is the default.

NL
indicates the new line character (X'15') should be used to delineate lines when reading or writing a BFS file.

lrecl
indicates the file should be treated as a fixed file, with no interpretation of records based on end of line characters. When BFSLINE *lrecl* is in effect, the file is presented as a fixed record format (RECFM=F) file with a logical record length (LRECL) equal to the *lrecl* value. No end of line characters are removed from or added to the file when it is read from or written back out to the byte file system if BFSLINE *lrecl* is still in effect.

The last record will be padded with blanks if *lrecl* is greater than 1 and the last record does not completely fill the last logical record.

CRLF
indicates carriage return and line feed characters (X'0D25') should be used to delineate lines when reading or writing a BFS file.

CRNL
indicates carriage return and new line characters (X'0D15') should be used to delineate lines when reading or writing a BFS file.

/string/
allows the user to specify 1-2 characters that are used to delineate lines when reading or writing a BFS file. These characters are translated to uppercase before they are utilized. The slash character (/) may not be one of these characters (in other words, you cannot enter ///).

/hexstring/
specifies a hexadecimal string of 2 or 4 characters that defines the value to be used for BFSLINE. The *hexstring* must be in the format X'nnnn' or X'nn'. You must not specify any spaces in the string, and there must be 2 or 4 hexadecimal characters in the string.

BFSLINE is not used unless NAMETYPE BFS is in effect.

NAMetype
Use the NAMETYPE option to specify whether the file ID will be interpreted to be in the form used for CMS record files or in the form used for byte file system (BFS) files.

CMS
Indicates file IDs will be interpreted to be CMS record files. They will be interpreted as *fn ft fm*. This is the default.

BFS
Indicates file IDs will be interpreted as path names (*pathname*). E for a description of the BFS path name syntax.

The following options are significant only if XEDIT is to be used in update mode, but they are ignored if NAMETYPE BFS is specified:

Seq8
specifies 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. The default value is SEQ8.

NOSeq8
specifies 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 *fn1* CNTRL is an update control file that controls the application of multiple update files to the file to be edited. If the control file is empty, an error message is displayed. (For more information on the CMS UPDATE command, see “UPDATE” on page 1092.)

This option automatically forces the UPDATE and SEQ8 options.

A maximum of 32 unique AUX file names may be specified in *fn1* CNTRL. If an auxiliary control file is empty, an informational message is displayed.

NOCtl
specifies the editor is not to use the control (CTL) file (even if it is specified in the LOAD subcommand in the profile).

Merge
specifies all the updates made through the control file and all the changes made while editing will be written into the file whose name is defined by the latest update level (that is, the most recently applied UPDATE file in a control file). This option forces the UPDATE option.

Note: The update file resulting from the merge may have different sequence numbers than if the updates were applied individually.

UNtil *ft1*
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 file type specified with the UNTIL option.

With the UNTIL option, you can specify file types of update files listed in the control file or of update files listed in an auxiliary control file. Do not specify AUX file types (AUX*xxxxx*) with the UNTIL option.

The UNTIL option forces the UPDATE option.

Incr *incr*
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 this option is not specified, the minimum increment is 1. This option forces the UPDATE option.

SIDcode *string*
specifies a string the editor inserts in every line of an update file, whether the update file is being created or is an existing file. The editor inserts the specified string in the first eight columns of the last 17 columns of the file line (lrecl-16 to lrecl-9). For example, if you have a file with fixed, 80-character lines, the editor inserts the string in columns 64 through 71. If the string is fewer 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. To use XEDIT on a file in an SFS directory, the directory must be accessed and you must have authorization to access the file. For a file in an SFS file control directory, you must have either read or write authority. For files in SFS directory control directories, you must have either directory read or directory write authority for the directory in which the file resides. For more information on SFS directories and authorizations, see *z/VM: CMS User's Guide*.

   To use XEDIT on a BFS file, you must have permission to access the file. For more information on the OPENVM PERMIT command, see *z/VM: OpenExtensions Commands Reference* or enter HELP OPENVM PERMIT.
2. When the LOCK option is in effect for an SFS file, it is possible for the lock to be removed during your edit session if one of the following abnormal errors occurs:
   - File pool server failure
   - Network or Advanced Program-to-Program Communications/VM (APPC/VM) failure on the last data link with the file pool server
   - Your virtual machine abends or is re-IPLed

   Note: If an abend occurs, the update session lock obtained by XEDIT will be deleted if sufficient storage is available for this additional processing.
3. To change files in an SFS directory control directory, you must access the directory in read/write mode. Because CMS lets only one user at a time access an SFS directory control directory in read/write mode, no one can change a file while you are editing it.

   You can use XEDIT on files within a directory control directory that is accessed in read-only mode. No warning message is displayed, but because the directory is accessed in read-only mode, you cannot

change the copy of the file in the directory by using XEDIT subcommands such as FILE and SAVE. To save the changes, you must save or file the file on a minidisk or directory you have accessed in read/write mode.

To change files in a file control directory, you can access the directory in either read-only or read-write mode. (By default, when you access someone else's directory, the access mode is read-only.) To have XEDIT respect the read-only access for file control directories regardless of your file authority, use the CMS SET RORESPECT ON command. (See "SET RORESPECT" on page 987 for an explanation of this command.)

4. The CMS RORESPECT setting may be used to prevent XEDIT from writing to a file control directory accessed read-only. To prevent XEDIT from locking a file in the directory so others may write to it, you may query the CMS setting and issue a LOAD subcommand with the NOLOCK option from your profile.
5. If you XEDIT an SFS or BFS file that has been migrated (moved to DFSMS/VM-owned storage), it is automatically recalled if the CMS SET RECALL command is set to ON. For more information, see "SET RECALL" on page 980. This may cause a slight delay before the file can be accessed. If CMS SET RECALL is OFF the file will not be recalled and an error message will be displayed.
6. For the PROFILE, CTL, SIDCODE, INCR, UNTIL, MEMBER, WIDTH, WINDOW, BFSLINE, and NAMETYPE options, the operand must be specified; otherwise, the next option is interpreted as its operand. For example, in the PROFILE *macroname* option, *macroname* must be specified; if it is not, the next option is interpreted as the operand *macroname*.
7. Once the XEDIT *command* has been executed, the XEDIT *subcommand* can be used to edit and display multiple files simultaneously (see the XEDIT subcommand in *z/VM: XEDIT Commands and Macros Reference*).
8. You can also call the editor recursively (that is, using the CMS XEDIT command). This ability is particularly useful when applications are developed using the editor and its macro facilities to interface with the user, for example, HELP.

   When you call CMS XEDIT recursively, a new ring of files is begun that is independent of any previous ring(s).
9. The MEMBER option and the NOUPDATE option have no effect when preceded by an option that automatically forces update processing. Also, options that usually force update processing are ignored when the MEMBER option or the NOUPDATE option precedes them.
10. If full-screen CMS is set ON before XEDIT writes to the screen, XEDIT issues the WINDOW SHOW CMSOUT command followed by WINDOW SHOW XEDIT or a WINDOW SHOW for the particular window that has been set up to display the file.
11. 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).
12. When an XEDIT command invokes the PROFILE macro, everything following the command name XEDIT is tokenized (truncated to eight characters), and then assigned to the argument string passed to the PROFILE macro as the first parameter.

    The editor does not examine any parameters that follow a closing right parenthesis on the XEDIT command.

    When you specify the NAMETYPE BFS option on the XEDIT command, your profile is called as a REXX function, and the untokenized, mixed case file ID (shown below as *fileid2*) is passed unchanged as a second argument string.

    Note: The second parameter (*fileid2*) is still enclosed in quotation marks if that is the way it was entered.

    This is useful when editing a BFS file because the path name may be up to 1023 bytes long, is case sensitive, and may contain blanks and other special characters, such as quotation marks. The XEDIT profile must be in REXX, not in EXEC or EXEC2, when the NAMETYPE BFS option is used. The return code is set to **RC=5** if the profile returns a result that is not valid.

Whether or not NAMETYPE BFS is specified, special substitution is done for a file ID enclosed in quotation marks. It is assumed this is a BFS path name containing special characters such as blanks or a (, and a place holder, the character '*', is substituted in its place in the argument list for the first argument string so the profile option processing works correctly.

An example of a way to obtain these argument strings from within your profile when the NAMETYPE BFS option is used is:

```
PARSE ARG fileid1 ft fm '(' OPTIONS , fileid2
```

In the example above, *fileid1* can be:

- The file name for a CMS file
- A path name for a BFS file that does not contain any blanks or special characters
- A placeholder for a file ID enclosed in quotation marks on the command invocation

13. When you enter an XEDIT command for an SFS or minidisk variable-format file, trailing blanks are removed when it is filed (or saved).

    When the record format is variable (V) for a BFS file, trailing blanks are removed when it is filed (or saved), and a line containing all blanks is represented in the file as two BFSLINE values in a row. (The initial record format setting for a BFS file depends upon the BFSLINE option. For the BFSLINE option description, see the BFSLine option. on the XEDIT command for more information.
14. Update control comment records cannot be edited when the UPDATE option is in effect. Edit the update file without specifying any update options to change these records.
15. Many languages have more characters than can be displayed using one-byte codes (KANJI, for example). A Double-byte Character Set (DBCS) represents these characters. The double-byte characters can appear in a sentence with characters from other languages that are displayed in 1-byte codes. Files containing double-byte characters are handled differently from files that contain only 1-byte characters.

    When specifying a BFSLINE value for use on files containing DBCS characters, ensure you use a value that will not conflict with DBCS characters. The hexadecimal code for a DBCS character must be X'0000', X'4040', or X'*aabb*', with *aa* and *bb* both being in the range of X'41' to X'FE'.
16. The virtual storage required to edit a ring of one or more files is greater than the size of the files themselves. This includes storage necessary to manage the ring, the lines, the display, and other internal XEDIT storage needs.
17. Unless BFSLINE *lrecl* is in effect when you XEDIT a BFS file, a blank line is inserted if two BFSLINE values are found in a row.
18. The BFSLINE setting has no effect when a PUT, PUTD, GET, FILE, SAVE, or LOAD subcommand reads or writes a CMS record file when NAMETYPE CMS is in effect.
19. The BFSLINE option can also be used on the XEDIT subcommand from within an XEDIT session. For example, a user can enter:

    ```
    XEDIT a a a
    ```

    and then enter

    ```
    XEDIT pathname (BFSLINE NL NAMETYPE BFS
    ```

    on the XEDIT command line.
20. Commit behavior of BFS files is different from that of SFS or minidisk files. For example, if the editor encounters an error when writing to a minidisk or SFS file, your changes are reversed and the original file is preserved. This is not the case for BFS files. The editor creates a copy of the file in XEDTEMP CMSUT1 on your A disk so you can recover your data if an error occurs when writing the file to the byte file system.

    If the editor is unable to create the XEDTEMP CMSUT1 file, you will receive a message describing the problem, and a second message:

```
595E  Not able to create CMS file used for recovery. Correct
      error or QQUIT to exit without writing file
```

If the editor encounters an error while writing the byte file system file such that the contents of an existing file are damaged, you will receive a message describing the problem, and a second message to tell you that you must take action to recover your file.

```
024E  File XEDTEMP CMSUT1 A1 contains file contents; use
      OPENVM PUT to recover file
```

21. When a new BFS file is created, the owning UID established is the effective UID of the VM User ID on which the request was issued. The GID is the GID of the parent directory. Use OPENVM OWNER to change the defaults. Refer to *z/VM: OpenExtensions Commands Reference* or enter HELP OPENVM OWNER for more information.
22. Permissions for a new BFS file are set based on the current value of the mask, with the exception of execute permissions, which are not set to ON. Use OPENVM PERMIT to change the defaults. For more information on these OPENVM commands, see *z/VM: OpenExtensions Commands Reference* or enter HELP OPENVM.
23. You can XEDIT path names only if they represent BFS regular files.
24. Typically you would use the default BFSLINE option when XEDITing a file in an NFS-mounted file system.

    If the NFS-mounted file system is a minidisk or SFS directory, you must coordinate XEDIT's BFSLINE option with the VMNFS server options specified on the mount. This is necessary because the NFS-mounted file is changed into byte-stream format when read by XEDIT, and changed back into record format when written to the minidisk or SFS directory through NFS.

    If the remote CMS file is a Variable file, you could use *lines=NL* on the mount and the default BFSLINES NL in the XEDIT options. If the remote CMS file is a Fixed file, you could specify *lines=none* in the mount options and BFSLINES *lrec*l in the XEDIT options.

Responses

When editing a file that resides in an SFS FILECONTROL directory, if you have only read authority to the file and you do not specify the NOLOCK option, you receive the message:

```
1299W  Warning: not authorized to lock fn ft fm
```

The editing session continues but the file is not locked.

The following messages are displayed only if you are using XEDIT in update mode:

```
178I  Updating fn ft fm
      Applying fn ft fm
1229I fn ft fm is empty
            .
            .
            .
180W  Missing PTF file fn ft fm
```

If the XEDIT work file, XEDTEMP CMSUT1, exists on a file mode accessed R/W as a result of a previous edit session that ended abnormally, you will receive the message:

```
024E   File XEDTEMP CMSUT1 fm already exists
```

You can use the CMS TYPE command to examine the existing file. If you decide you want to keep it, use the CMS RENAME command to give it a new CMS file ID, or use OPENVM PUTBFS to move it into the byte file system. Refer to *z/VM: OpenExtensions Commands Reference* or enter HELP OPENVM PUTBFS for more information on this command. If the file is incorrect or incomplete, erase it and enter the command again.

If you try to XEDIT an existing empty file and a profile is **not** executed, you get the following message:

```
556I  Editing existing empty file:
```

Messages and Return Codes

- DMS002E File *fn ft fm* not found [RC=28]
- DMS003E Invalid option: *option* [RC=24]
- DMS024E File XEDTEMP CMSUT1 *fm* already exists [RC=28]
- DMS029E Invalid parameter *parameter* in the option *option* field [RC=24]
- DMS033E File is not a regular BFS file [RC=32]
- DMS048E Invalid filemode *mode* [RC=24]
- DMS054E Incomplete [or incorrect] fileid specified [RC=24]
- DMS062E Invalid character in fileid {*fn ft fm*|*pathname*} [RC=20]
- DMS065E *option* option specified twice [RC=24]
- DMS066E *option1* and *option2* are conflicting options [RC=24]
- DMS069E Filemode *mode* not accessed [RC=36]
- DMS070E Invalid parameter *parameter* [RC=24]
- DMS104S Error *nn* reading file *fn ft fm* from disk or directory [RC=31, 55, or 100]
- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS132S File [*fn ft fm*] too large[: *pathname*] [RC=88]
- DMS137S Error *nn* on STATE for *fn ft fm* [RC=88]
- DMS229E Unsupported OS dataset, error *nn* [RC=80, 81, 82, or 83]
- DMS500E Unable to unpack file *fn ft fm* [RC=88]
- DMS508E LOAD must be the first subcommand in the profile [RC=3]
- DMS512E This is not allowed in CMS subset mode [RC=100]
- DMS554E Not enough virtual storage available [RC=104]
- DMS556I Editing existing empty file:
- DMS571I Creating new file:
- DMS622E Insufficient free storage (for {MSGLINE|PFkey/PAkey|synonyms})
- DMS915E Maximum number of windows already defined [RC=13]
- DMS927E The virtual screen must contain at least 5 lines and 20 columns [RC=24]
- DMS928E Command is not valid for virtual screen CMS [RC=12]
- DMS1019E Network File System name is not allowed
- DMS1020E Foreign host cannot be reached. The request returned return code *rc* and reason code *rs*
- DMS1138E File sharing conflict for file {*fn ft fm*|*pathname*] [RC=70]
- DMS1214W File *fn ft fm* already locked SHARE
- DMS1215E File *fn ft fm* is locked or in use by another user [RC=70]
- DMS1229E *fn ft fm* is empty [RC=88]
- DMS1229I *fn ft fm* is empty
- DMS1262S Error *nn* opening file *fn ft fm* [RC=31, 55, 70, 76, 99, or 100]
- DMS1299W Warning: Not authorized to lock file *fn ft fm*
- DMS1300E Error *nn* {locking|unlocking} file *fn ft {fm|dirname}* [RC=55, 70, 76, 99, or 100]
- DMS2105E Permission is denied [24]

- DMS2134E Return code *bpxrc* and reason code *bpxrs* given on call to *rtnname* [for path name *pathname*] [RC=100]
- DMS2154E File {*fn ft fm*|*pathname*} is migrated and implicit RECALL is set to OFF [RC=50]
- DMS2155E DFSMS/VM error occurred during creation or recall of file {*fn ft fm*|*pathname*} [RC=51]
- DMS2526E File or directory creation or file recall was rejected by a DFSMS/VM ACS routine; ACS routine return code *acs rcode* [RC=51]

Messages with Member Options:

- DMS007E File *fn ft fm* is not fixed, 80-character records [RC=32]
- DMS033E File *fn ft fm* is not a library [RC=32]
- DMS039E No entries in library *fn ft fm* [RC=32]
- DMS167S Previous MACLIB function not finished [RC=88]
- DMS622E Insufficient free storage for reading map [RC=104]

Messages with Update Option:

- DMS007E File *fn ft fm* is not fixed, 80-character records [RC=32]
- DMS007E File *fn ft fm* does not have a logical record length greater than or equal to 80 [RC=32]
- DMS007E File *fn ft fm* does not have the same format and record length as *fn ft fm* [RC=32]
- DMS007E File *fn ft fm* is not fixed record format [RC=32]
- DMS104S Error *nn* reading file *fn ft fm* from disk or directory [RC=31, 32, or 55]
- DMS174W Sequence error introduced in output file: *seqno1* to *seqno2* [RC=32]
- DMS178I Applying *fn ft fm*
- DMS179E Missing or invalid MACS card in control file *fn ft fm*
- DMS180W Missing PTF file *fn ft fm*
- DMS183E Invalid {CONTROL|AUX} file control card [RC=32]
- DMS184W ./ S not first card in update file--ignored [RC=32]
- DMS185W Non numeric character in sequence field *seqno* [RC=32]
- DMS186W Sequence number not found [RC=32]
- DMS207W Invalid update file control card [RC=32]
- DMS210W Input file sequence error: *seqno1* to *seqno2* [RC=32]
- DMS317E Number of AUX file types in control file *fn ft fm* exceeds 32 [RC=32]
- DMS570W Update *ft* specified in the UNTIL option field not found
- DMS597E Unable to merge updates containing ./ S cards [RC=32]
- DMS1262S Error *nn* opening file *fn ft fm* [RC=31, 55, 70, 76, 99, or 100]

Return codes:

RC
Meaning

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 a 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
A character in the file name, file type, or path name is not valid

24
Not valid 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 maximum number of AUX file types exceeded, or BFS file is not a regular file, or Network File System path name cannot be used

36
Corresponding minidisk or directory not accessed

50
File is DFSMS/VM migrated and automatic recall has been set to OFF (CMS SET RECALL command)

51
DFSMS/VM error

55
APPC/VM communications error or TCP/IP communications error

70
File sharing conflict or the minidisk file being opened is already open using CSL interfaces of DMSOPEN or DMSOPDBK

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

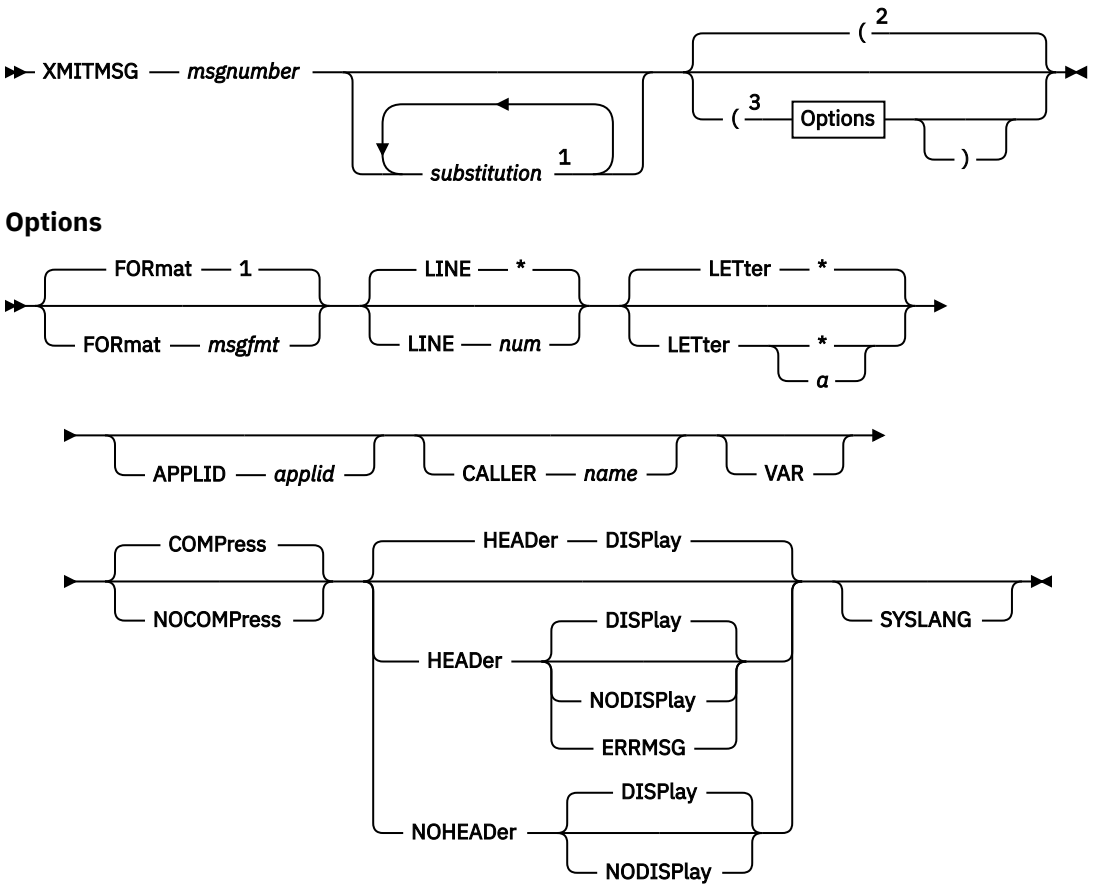
88
File is too large and does not fit into storage, a previous MACLIB function was not finished, or an unsupported function was attempted with an empty file

99
A required system resource is not available

100
Error reading the file into storage

104
Insufficient storage available

XMITMSG



Notes:

[1] A maximum of 20 substitutions is allowed.
[2] The defaults you receive appear above the line in the Options fragment.
[3] You can enter Options in any order between the parentheses.

Authorization

General User

Purpose

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 you requested. XMITMSG can be used in a REXX, EXEC 2, or CMS environment.

Operands

msgnumber
is a 1-4 digit number used to locate its associated message text in the repository.

substitution
specifies the substitutions to be done on the message.

If XMITMSG is issued from the CMS command line, use single quotation marks around the items in the *sublist* to indicate literal substitution; numeric values not in single quotation marks will then be treated as dictionary substitutions and will be retrieved from the repository. Literal substitutions must not contain blanks or parentheses.

If XMITMSG is issued from an exec, you will already be using one kind of quotation mark to surround the call to XMITMSG. Use the opposite kind to indicate literal substitution in the *sublist*. (For example, if you used double quotation marks around the XMITMSG call, use single quotation marks to indicate literal substitution.)

If the dictionary substitution specified in the *sublist* is not found in the repository, '*nnnn*' will be substituted for it.

A maximum of 20 substitutions is allowed.

Options

FORmat *msgfmt*
specifies a 1-2 digit format number. This identifies the different versions of the same message which have the same message number. The numbering of formats is from 01 to 99. The default is 1. A format of 00 is not allowed.

LINE *num*
LINE *
specifies a 1-2 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 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*
LETter *
specifies the severity letter of the message. Message severity is provided already within the message repository module, and this letter is returned when you specify *. This is the default. Specify a letter for *a* if 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 the first three characters of the exec name (if these characters are different from the application ID), or else the next three characters of the exec name. If the CALLER exec's name is more than three characters long, it is truncated.

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 variables MESSAGE.nn, with the first line in MESSAGE.1, and second line in MESSAGE.2, and so forth. The number of lines in the message is copied into MESSAGE.0.

To display the message on the terminal and also return the message to the exec variables, you must specify the DISPLAY option and the VAR option. Otherwise, if you just specify the VAR option, the default displaying option is NODISPLAY and the message is not displayed at the terminal.

If ERRMSG is specified, the message header and message text returned in MESSAGE.nn is determined by the CP EMSG setting.

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.

If a message is defined in the repository with substitution indicators (&1, &2, and so forth) and the message is invoked by XMITMSG with no substitution specified in the *sublist* and without the NOCOMPRESS option, blanks are substituted for the substitution indicators when the message text is scanned. With COMPRESS specified (or as the default option), the blanks are removed. If the substitution indicator in the message text is not an '&' followed by an integer, XMITMSG will treat it as part of the message text, and will not substitute a blank for it.

HEADer
Specifies the message header is created for the message. This is the default.

The message header consists of:

```
xxxmmmnnns
```

or

```
xxxmmmnnnns
```

Where:

xxx
specifies the application ID, *DMS* for CMS

mmm
specifies the CALLER name

nnn* or *nnnn
specifies the message number

s
specifies the severity code

These are the most commonly used severity codes:

Code
Message Type

E
Error

I
Information

R
Response

S
Sever

T
Terminal

W
Warning

If NODISPLAY is specified along with HEADER, the NODISPLAY option overrides the HEADER option, and no message is displayed.

Also, if VAR is specified, the message header and message text is returned in MESSAGE.*nn* (even if NODISPLAY is specified).

If ERRMSG is specified, the HEADER option is ignored and the message is processed according to the CP EMSG setting.

NOHEADer
Specifies the message header is not displayed on the terminal. Also, if VAR is specified, the message header is not returned in MESSAGE.nn. You may not specify this option with the ERRMSG option.

DISPlay
Specifies the message text is displayed on the terminal, regardless of the CP EMSG setting. This is the default option unless you specify VAR. If VAR is specified, the default display option is NODISPLAY.

Do not specify this option with the ERRMSG option.

NODISPLay
Specifies the message text is not displayed on the terminal, regardless of the CP EMSG setting. You may not specify this option with the ERRMSG option. This is the default option when you specify VAR.

ERRMSG
Specifies the message line is displayed according to the CP EMSG setting.

CMS always recognizes EMSG settings for all error (E), information (I), and warning (W) messages, but ignores the EMSG setting and displays the complete message (error code and text) for all response (R), severe error (S), and terminal (T) messages. If EMSG is set to:

- ON - the entire message is displayed, header plus text
- OFF - the message is not displayed
- TEXT - only the text portion is displayed
- CODE - only the header is displayed

ERRMSG overrides the HEADER option. You cannot specify ERRMSG with the NOHEADER, DISPLAY, or NODISPLAY options.

SYSLANG
specifies the system default language issues 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. For more information on how to create your own message repository, see *z/VM: CMS Application Development Guide*.
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. For more information on how to issue messages from assembler programs, see the APPLMSG macro in *z/VM: CMS Macros and Functions Reference*.
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.

6. Table 58 on page 1274 summarizes the interaction between the XMITMSG options entered and the format of the resulting output.

Table 58. XMITMSG Options and Resulting Output Format

Options Specified				Output Results			
DISPLAY	**HEADER**	**VAR**	**EMSG**	**Header Displayed**	**Text Displayed**	**Header Returned**	**Text Returned**
N/S	N/S	N/S	–	Y	Y	N	N
N/S	N/S	VAR	–	N	N	Y	Y
N/S	HEAD	N/S	–	Y	Y	N	N
N/S	HEAD	VAR	–	N	N	Y	Y
N/S	NOHEAD	N/S	–	N	Y	N	N
N/S	NOHEAD	VAR	–	N	N	N	Y
DISP	N/S	N/S	–	Y	Y	N	N
DISP	N/S	VAR	–	Y	Y	Y	Y
DISP	HEAD	N/S	–	Y	Y	N	N
DISP	HEAD	VAR	–	Y	Y	Y	Y
DISP	NOHEAD	N/S	–	N	Y	N	N
DISP	NOHEAD	VAR	–	N	Y	N	Y
NODISP	–	N/S	–	N	N	N	N
NODISP	N/S	VAR	–	N	N	Y	Y
NODISP	HEAD	VAR	–	N	N	Y	Y
NODISP	NOHEAD	VAR	–	N	N	N	Y
ERRMSG	HEAD"6.c" on page 1275	N/S	ON	Y	Y	N	N
ERRMSG	HEAD"6.c" on page 1275	N/S	CODE	Y	N	N	N
ERRMSG	HEAD"6.c" on page 1275	N/S	TEXT	N	Y	N	N
ERRMSG	HEAD"6.c" on page 1275	N/S	OFF	N	N	N	N
ERRMSG	HEAD"6.c" on page 1275	VAR	ON	Y	Y	Y	Y
ERRMSG	HEAD"6.c" on page 1275	VAR	CODE	Y	N	Y	N
ERRMSG	HEAD"6.c" on page 1275	VAR	TEXT	N	Y	N	Y
ERRMSG	HEAD"6.c" on page 1275	VAR	OFF	N	N	N	N

Table 58. XMITMSG Options and Resulting Output Format (continued)

Options Specified				Output Results			
DISPLAY	**HEADER**	**VAR**	**EMSG**	**Header Displayed**	**Text Displayed**	**Header Returned**	**Text Returned**

Notes on the Table:

a. A '-' indicates the option can be at any setting.

b. An 'N/S' indicates the option was not specified.

c. This is the default value.

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:

```
DMS875E Attempt to divide by 0 is invalid
```

Note: The variable DIVR must be included *inside* the quotation marks.

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:

```
DMS875E 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 */
  SAY MESSAGE.I         /*  message lines             */
END
```

displays the message:

```
DMS877E 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:

```
DMS877E to keep the return codes lined up on the next line.
```

Responses

Messages DMS813E and DMS814E can be displayed (depending on what you have specified for the ERRMSG, DISPLAY/NODISPLAY, and VAR options) if XMITMSG encounters an error when it attempts to retrieve the requested message.

A return code of 4 indicates the message text was truncated because after substitutions were made, it was longer than 240 characters.

Messages and Return Codes

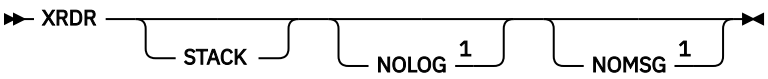
- DMS065E *option* option specified twice [RC=24]
- DMS066E *option1* and *option2* are conflicting options [RC=24]
- DMS080E Invalid *numtype* number [RC=24]
- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS405E Invalid or missing message number [RC=24]
- DMS408E Number of substitutions exceeds 20 [RC=24]
- DMS631E XMITMSG must be invoked from an EXEC 2 or REXX exec or as a CMS command [RC=24]

- DMS813E *repos* repository not found, message *nnnn* cannot be retrieved [RC=16]
- DMS814E Message number *nnnn*, format *nn*, line *nn*, was not found; it was called from *routine* in application *applid* [RC=12]
- DMS2045E Invalid substitution value - blank or parenthesis [RC=24]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

XRDR



Notes:

[1] If NOLOG and NOMSG are both specified, the message is suppressed and the reader file is not purged.

Authorization

General User

Purpose

Use the XRDR command to determine the characteristics of the next file in your virtual reader. XRDR generates a return code and either displays or stacks a message for each type of file recognized. RMSG or MAIL files are displayed on the screen and their contents are logged in a file called MAIL LOG. They are then purged from the reader.

The XRDR command examines the first file in the virtual reader, and stacks or prints a line describing the file ID, method to use for loading, type of file, and so on. Only files of the current reader spool class are examined. XRDR also handles files with a file type of MAIL (MAIL files).

Options

STACK
causes the information that would have been displayed to be stacked (typically for access by an exec). For RMSG or MAIL files, however, only a single descriptive line is stacked, rather than the complete message text.

NOLOG
prevents the logging of RMSG or MAIL files in a mail log file on disk. The message is displayed, but no further action is taken, and the file is purged from the reader. If NOLOG is used with the NOMSG option, the message is suppressed and the reader file is not purged.

NOMSG
suppresses the normal display of all text, including RMSG and MAIL files. If NOMSG is used with the NOLOG option, logging of RMSG or MAIL files is suppressed and the reader file is not purged. For RMSG or MAIL files, logging still occurs (if not suppressed by the NOLOG option).

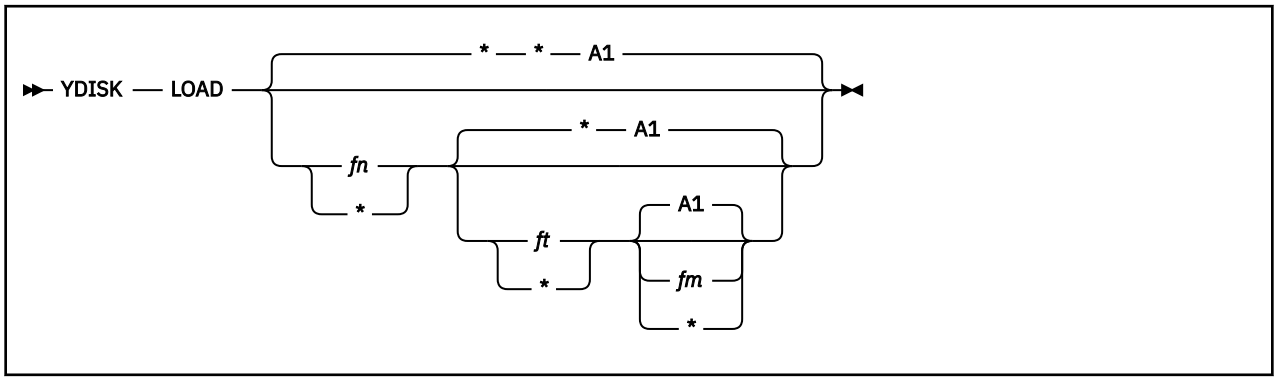
Usage Notes

1. If called without options, information regarding the first nonheld reader file of the active class is displayed. This requires spooling the reader with HOLD, reading the first record of the spool file, closing the reader, and spooling the reader NOHOLD, which may change the state of the virtual reader before executing XRDR. If the file is an RMSG file (first line beginning X'FE',C‘MSG:’) or punch file with a file type of MAIL, the complete file is displayed and then purged from the reader. The text of the message is also logged in the MAIL LOG A0 file unless the NOLOG option is given. All file types other than RMSG or MAIL remain in the reader after the command is complete.
2. Do not change the logical record length of the MAIL LOG A0 file. It must have a logical record length of 80 for XRDR to log RMSG and MAIL files.
3. In addition to the XRDR module, you can also use the preferred RDR command to obtain similar results. See the “RDR” on page 784 for details.

Messages and Return Codes

- DMS124S Error reading card file [RC=12|13]
- DMS205W No files in your reader [RC=0]
- DMS630S Error accessing spool file [RC=12]
- DMS2184I Printer file (*spoolid*) item length=*num*; Origin: *tag* [RC=2]
- DMS2184I Disk load *fn ft* (*spoolid*) Origin: *tag* [RC=3]
- DMS2184I Readcard *fn ft* (*spoolid*) Origin: *tag* [RC=4]
- DMS2184I Cards for IPL (*spoolid*) Origin: *tag* [RC=5]
- DMS2184I Unnamed card deck (*spoolid*) Origin: *tag* [RC=6]
- DMS2184I *type* file (*spoolid*) Origin: *tag* [RC=13]
- DMS2184I Unknown type (*spoolid*) Item length=*num*; Origin: *tag* [RC=15]
- DMS2184I VAFP printer file (*spoolid*) [RC=38]
- DMS2185I Active reader class empty (Next file *spoolid* Class *class*). [RC=1]
- DMS2186E Error *rc* from CP SPOOL READER HOLD. [RC=12]
- DMS2186E Error *rc* from CP SPOOL READER NOHOLD. [RC=12]
- DMS2186E Error *rc* from CP CLOSE READER. [RC=12]
- DMS2187E Error *rc* on *command*, logging terminated. [RC=12]
- DMS2188I Following *type* added to: *fn* Log *fm*. [RC=12]
- DMS2188I *type* logged in: *fn* Log *fm*.
- DMS8005E File with X'5A' CCW [RC=10]
- DMS8732E Reader not operational [RC=9]

YDISK



Authorization

General User

Purpose

The YDISK command is an alternative to the CMS DISK LOAD command. YDISK allows the name of the file to be changed when loading the file.

Operands

LOAD
loads a file from the spooled card reader that has been DISK DUMPed and writes it as a CMS file on the specified disk or directory.

fn ft fm
specify the new file name, file type, and file mode to be assigned the loaded file. Any of these may be coded as asterisk (*) to use the default values contained in the dumped file (the file mode defaults to A1). No checking is done to determine if the given file ID already exists.

Usage Notes

1. If the spool file contains multiple logical files and you specify a new file name or file type, YDISK only loads the first file, holds the spool file, and issues a message that explains the situation.
2. If the virtual reader is spooled continuous and a new file name or file type is specified, YDISK only loads the first file in the reader and issues a message indicating it has only loaded one file. See usage note 1 for information about loading spool files that contain multiple logical files.
3. In addition to the YDISK module, you can also use the preferred RECEIVE or DISK commands to obtain similar results. See “RECEIVE” on page 806 or “DISK” on page 178 for more details.

Messages and Return Codes

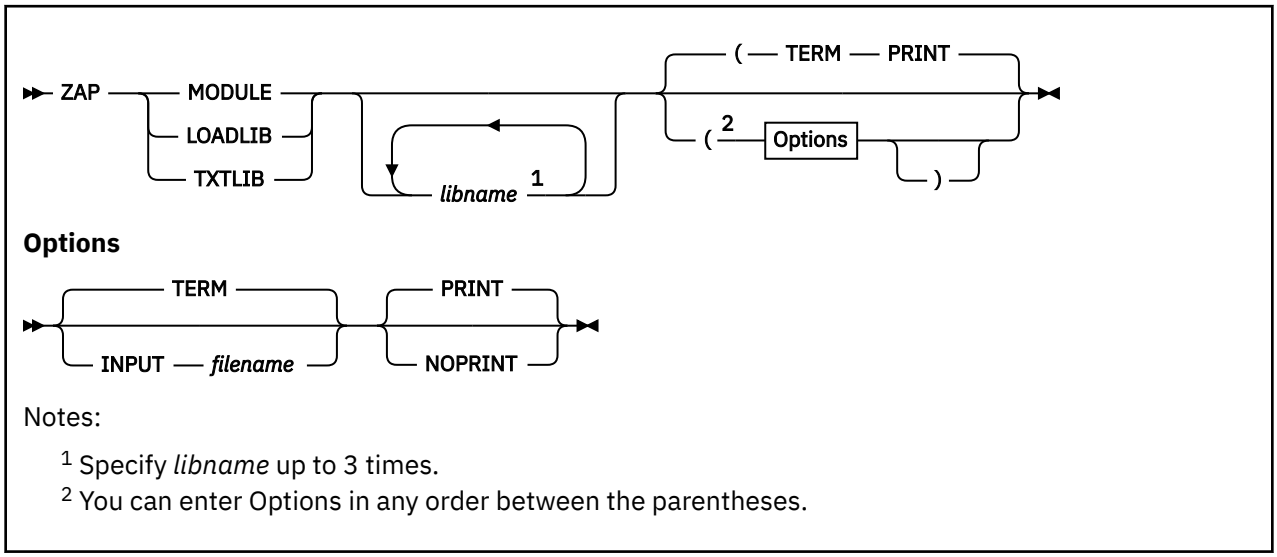
- DMS002E File *fn* not found [RC=28]
- DMS014E Invalid function *function* [RC=24]
- DMS037E Filemode *fm* is accessed as read/only [RC=36]
- DMS047E No function specified [RC=24]
- DMS054E Incomplete fileid specified [RC=24]
- DMS062E Invalid * in fileid *fn* [RC=20]
- DMS069E Filemode *fm* not accessed [RC=36]
- DMS070E Invalid parameter *parameter* [RC=24]

- DMS077E End card missing from input deck [RC=32]
- DMS078E Invalid card in input deck [RC=32]
- DMS104S Error *nn* reading file *fn ft fm* from disk or directory [RC=100]
- DMS105S Error *nn* writing file *fn ft fm* on disk or directory [RC=100]
- DMS109S Virtual storage capacity exceeded [RC=104]
- DMS118S Error punching file [RC=100]
- DMS124S Error reading card file [RC=100]
- DMS205W Reader empty, reader not ready or empty reader file [RC=8]
- DMS496S Invalid fileid '*fn ft fm*' found in input record [RC=100]
- DMS671E Error loading file *fn ft fm*; rc=*rc* from RENAME [RC=100]
- DMS1124W Spool file *spoolid* has been left in your reader because one or more files were not received [RC=1]
- DMS1215E File *fn ft fm* is locked by another user [RC=100]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

ZAP



Notes:

[1] Specify *libname* up to 3 times.
[2] You can enter Options in any order between the parentheses.

Authorization

General User

Purpose

Use the ZAP command to change or dump MODULE, LOADLIB, or TXTLIB files. You can use ZAP to change either fixed or variable length MODULE files. These files must be inactive (not preopened) at the time ZAP is invoked or called.

Input control records control ZAP processing. They can be submitted either from the terminal or from a CMS file. Using the VER and REP control records, you can verify and replace data or instructions in a control section (CSECT). Using the DUMP control record, you can dump all or part of a CSECT, an entire member of a LOADLIB or TXTLIB file, or an entire module of a MODULE file.

Operands

MODULE
LOADLIB
TXTLIB
are the choices of file type you can change or dump.

libname
is the file name of the library containing the member you want to change or dump. You can specify one to three library names. This operand is valid only for LOADLIB and TXTLIB files.

Options

TERM
indicates input is submitted through the terminal. After ZAP issues the prompting message ENTER:, you can enter input control records up to 80 characters long. This is the default.

INPUT *filename*
specifies input is submitted from a CMS file called *filename* ZAP. This file must be a fixed 80-byte sequential file residing on any file mode.

PRINT
specifies all output is printed on the printer. This is the default. If you specify PRINT with TERM, only error messages display at the terminal. If you specify PRINT with INPUT *filename*, only commands in error, control records in error, and error messages display at the terminal. See Table 59 on page 1283.

NOPRINT
specifies no output is printed on the printer. If you specify NOPRINT with TERM, all output except control records displays at the terminal. If you specify NOPRINT with INPUT *filename*, all output displays at the terminal. See Table 59 on page 1283.

Table 59. ZAP Command Options and Their Output

	PRINT	**NOPRINT**
INPUT	Everything prints on the printer. Commands in error, control records in error, and error messages display on the terminal.	Nothing prints on the printer. Everything displays on the terminal.
TERM	Everything prints on the printer. Error messages display on the terminal.	Nothing prints on the printer. Everything except control records displays on the terminal.

Input Control Records

There are eight types of ZAP control records:

- DUMP
- NAME
- BASE
- VER or VERIFY
- REP
- LOG
- COMMENT
- END

The ZAP program can accept only 80 characters of data for each control record. ZAP control records are free-form and need not start in position one of the record. Separate all information by one or more blanks. All address fields including the displacement (*disp*) fields in the VER and REP control records must contain an even number of hexadecimal digits, to a maximum of eight digits (such as X'0D', X'02C8', X'014318' and X'01234567'). Data fields in the VER and REP control records must also contain an even number of hexadecimal digits.

Note:

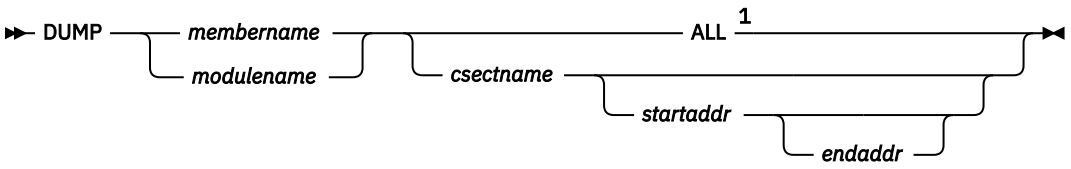
1. If you want, you can separate the data anywhere by commas (for example, 83256482 or 8325,6482). The commas have no effect on the operation.
2. Do not use blank spaces as separators **within** data fields.

ZAP sets the NOGO switch to ON if it finds a control record in error. A file cannot be changed if the NOGO switch is turned on. The next valid NAME record turns the NOGO switch off. This means that if the control record in error is the NAME record, all succeeding records are ignored until the next NAME, DUMP, or END record. For any other error, only REP control records that follow are ignored.

DUMP Control Record

The DUMP control record lets you dump a portion or all of a specified control section, or the complete member or module. The format of the output of the dump is hexadecimal with an EBCDIC translation of the hexadecimal data.

The DUMP control record is optional and resets the NOGO switch off. The DUMP control record must not immediately precede a BASE, VER, or REP control record. A NAME control record must precede the BASE, VER, and REP control records (if any) that follow a DUMP control record.



Notes:

[1] You can specify ALL for MODULE files, LOADLIBs, and OS TXTLIBs, but not for CMS TXTLIBs.

Parameters

membername
is the name of the member you want to dump, or the member that contains the CSECT(s) you want to dump. This member must be in one of the libraries you specify on the ZAP command.

For a CMS TXTLIB, the format of the dump control record requires you specify both *membername* and *csectname*. Because the library directory does not contain member names, any word can be used to replace *membername*. The program searches for only the second name following the DUMP statement; therefore, the second name must be *csectname*.

modulename
is the name of the module you want to dump, or the module that contains the CSECT(s) you want to dump. If you specify a module that has no loader table, the program dumps the entire module.

csectname
is the name of the control section you want to dump. If you do not specify *csectname*, the program dumps only the first CSECT.

This parameter is required for CMS TXTLIBs, but is optional for OS TXTLIBs, LOADLIBs, and MODULE files. For more information, see Name Control Record. You must not specify *csectname* for a module created with the NOMAP option.

startaddress
is the location within the specified CSECT where you want the dump to begin. This must be two, four, six or eight hexadecimal digits. The start address is the displacement from the beginning of the CSECT. For example, to start dumping at address X'08' in a CSECT that begins at location X'400', you specify start address X'08', not X'0408'.

endaddress
is the last address you want to dump. This must be two, four, six or eight hexadecimal digits. If you do not specify *endaddress*, the program dumps from *startaddress* to the end of the CSECT.

Note: The start and end addresses apply only when you specify *csectname*.

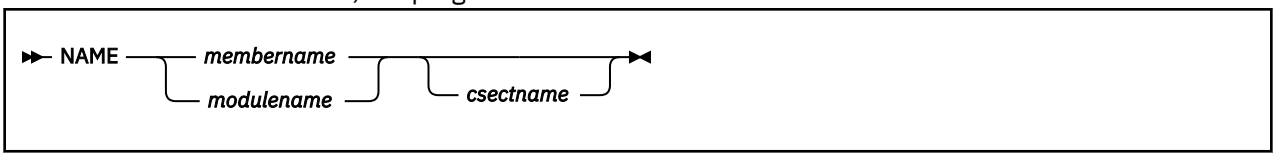
ALL
tells the program to dump all CSECTs within the member or module you specify. You can specify ALL for MODULE files, LOADLIBs, and OS TXTLIBs, but not for CMS TXTLIBs. To dump all the CSECTs in a member of a CMS TXTLIB, you must issue a separate DUMP control record for each CSECT.

Usage Notes

1. Displacements listed in the dump output for a module file are calculated from the beginning location of the module. Therefore, the addresses in the output might differ from the displacements within a CSECT.
2. If a DUMP control record references a TXTLIB file that contains ORG statements causing more than one occurrence of an address, data found at the first occurrence is displayed, but any subsequent redefinition of data for the same location is ignored.
3. The ZAP command does not support extended format module files created by the BIND command.

NAME Control Record

The NAME control record specifies the member or module and CSECT that contain the data you want the ZAP operation to verify or replace. The NAME control record must precede the BASE, VER, and REP control records. If it does not, the program sets the NOGO switch on.



Parameters

membername
is the member you want to search for the desired CSECT.

modulename
is the module you want to search for the desired CSECT.

csectname
is the name of the control section you want to change.

Usage Notes

1. You must specify *csectname* if the CSECT you want to change {*modulename*} is in a CMS TXTLIB (that is, a TXTLIB created by the TXTLIB command from CMS TEXT decks that do not have a NAME card following the END card). The directory of a CMS TXTLIB contains only CSECT names and no member names. Select a word to replace *membername* as the first entry following the NAME operand in the NAME statement for a CMS TXTLIB.

   Note: The word you specify for the *membername* for a CMS TXTLIB should be a meaningful name. The file name of the LOG control record is determined by the *membername* or *modulename* you specify in the NAME control record.
2. The CSECT name you specify in the NAME record is compared with CSECT names in the directory. If the CSECT(s) match and no member name match is found, the member selected is the one that contains the CSECT name.
3. The *csectname* is optional if the CSECT you want to change is a LOADLIB or an OS TXTLIB (that is, a TXTLIB created by the TXTLIB command from CMS TEXT decks that have a NAME card after the END card). The dictionaries of the specified libraries are searched for the member name, and the member is then searched for the CSECT name, if you specified one. If you do not specify *csectname* for a LOADLIB or an OS TXTLIB, the program uses the first control section.
4. The *csectname* is optional for a MODULE file. The module named in the NAME control record is found and, if you specify *csectname*, the first record is read to determine the number of records in the module and the availability of a loader table, which the program can then search for *csectname*. If you do not specify *csectname*, the program uses the beginning location of the module. You cannot specify *csectname* if the module was created with the NOMAP option.

BASE Control Record

The BASE control record adjusts displacement values for subsequent VER or REP control records for a CSECT whose starting address is not location zero in an assembly listing.

The BASE control record is optional. For more information, see the description of the VER control record. If you specify the BASE control record, it must follow the NAME record, but it need not follow the NAME record immediately. For example, you could have the following sequence of control records: NAME, VER, REP, BASE, VER, REP.

Parameter

address
is the starting address of the CSECT. It must be two, four, six or eight hexadecimal digits.

Usage Note

If you do not specify a *csectname* in the NAME control record, you cannot specify any BASE value other than 00.

Example

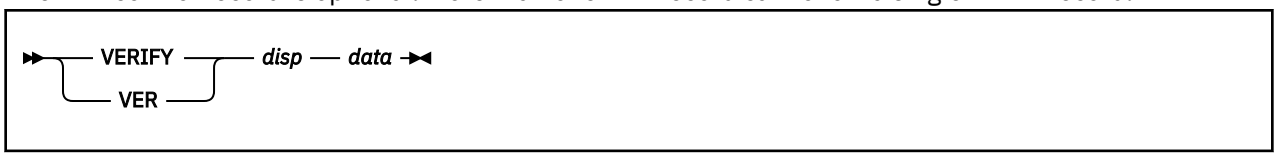
For a CSECT starting at location X'400', you specify BASE 0400 in the BASE control record. If a subsequent VER card requests verification of location X'0408', BASE 0400 is subtracted from X'0408', and the program verifies location X'08' in the CSECT. This example applies if you specify TXTLIB, LOADLIB, or MODULE and the module map is present.

However, if no module map is present for a MODULE file (that is, the module was generated with the NOMAP option), all operations are done as if the BASE address is location X'0'. For example, if you specify a BASE of X'400' and the address you want to look at or change is X'408', you must specify X'08' and not X'408' in the REP and VER control records. The address in this case is from the start of the module.

VER or VERIFY Control Record

The VER control record requests verification of instructions or data within a CSECT. If the verification fails, the program does not do a subsequent REP operation until it finds another NAME control record.

The VER control record is optional. More than one VER record can follow a single NAME record.



Parameters

disp
is the displacement from the start of the CSECT containing the data to be inspected, if you did not submit a BASE control record for this CSECT. The variable *disp* can also be the actual location of the data to be inspected, if you did submit a BASE control record. The variable *disp* must be two, four, six or eight hexadecimal digits. This displacement does not have to be aligned on a fullword boundary. If this displacement value is outside the limits of the CSECT specified by the preceding NAME control record, the VERIFY control record is rejected.

data
is the data against which the data in the CSECT is compared. This must be an even number of hexadecimal digits.

Usage Note

If the VER control statement references data in a TXTLIB file that is later redefined by ORG statements, only the first data definition is verified.

Example

If the location you want to verify is X'3CC', and the CSECT begins at location X'2B0', you can enter:

```
base 02B0
ver 03CC data
```

or you can omit the BASE control record, subtract the CSECT start address from the address of the data, and enter:

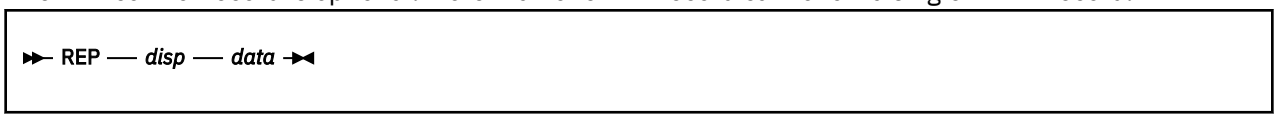
```
ver 011C data
```

REP Control Record

The REP control record changes instructions or data at the specified location within the CSECT you specify in a preceding NAME control record. The data specified in the REP control record replaces the data at

the CSECT location specified by the *disp* parameter. This replacement is on a one-for-one basis; that is, one byte of data defined in the control record replaces one byte of data at the location you specify. If the replacement fails, the program does not do additional REP operations until it finds another NAME control record.

The REP control record is optional. More than one REP record can follow a single NAME record.



Parameters

disp
is the displacement from the start of the CSECT of the data you want to replace (if you did not submit a BASE control record for this CSECT). Variable *disp* can also be the actual location of the data if you did submit a BASE control record. Variable *disp* must be two, four, six or eight hexadecimal digits. This displacement need not address a fullword boundary. If this displacement value is outside the limits of the CSECT being changed, the program does not do the replacement operation.

data
is the data that is to replace the data in the CSECT. This must be an even number of hexadecimal digits.

Usage Notes

1. Although you do not have to verify a location before replacing data, you should do so to make sure the data being changed is what you expect it to be.
2. If the REP control statement references data in a TXTLIB file that is later redefined by ORG statements, the replacement of data takes place at the first occurrence of the data address of the TXTLIB member.

Example

If the location you want to replace is X'3CC', and the CSECT begins at location X'2B0', you can enter:

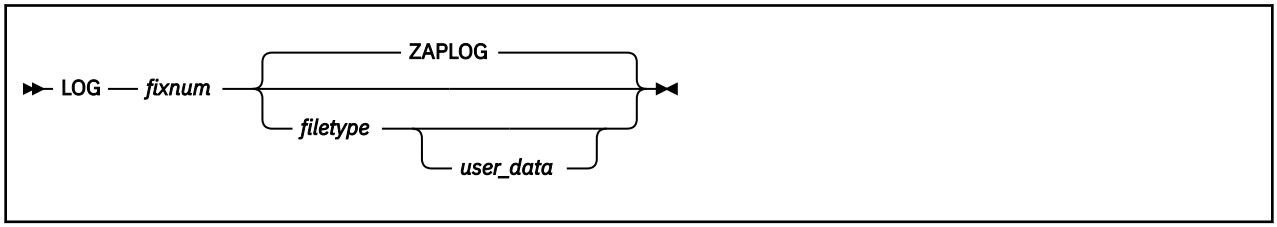
```
base 02B0
rep 03CC data
```

or you can omit the BASE control record, subtract the CSECT start address from the address of the data, and enter:

```
rep 011C data
```

LOG Control Record

The LOG control record lets you specify, after you apply a fix, a unique fix number that is recorded in a log file for the module or member. The file name of the log file is the same as *membername* or *modulename* in the NAME control record.



Parameters

fixnum
specifies the number associated with the fix. Its length may vary from one to eight alphanumeric characters.

filetype
specifies the file type of the log. The default is ZAPLOG.

user_data
specifies any data you want to enter into the log. If you specify *user_data*, you must specify *filetype*.

Usage Notes

1. The LOG control record is optional and is allowed only if valid NAME and REP control records are found. The file name is obtained by the log routine from *modulename* or *membername* in the NAME control record. However, if no LOG control record is found, a dummy log record is written at the end of the user's valid REPs.
2. Log multiple names by including a LOG control record after each name. If the LOG record is not included after each name, error message DMS070E results. Processing continues after the error messages occur.
3. The LOG record is 80 bytes in length and contains the following information:
   - Columns 1-63 contain the fixnum and, if specified, the file type and user data.
   - Columns 64-80 contain the date and time of the ZAP.

COMMENT Control Record

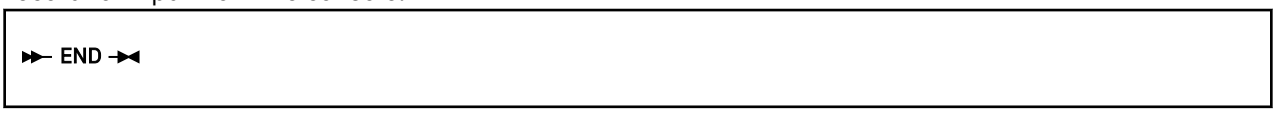
The ZAP program ignores COMMENT control records. If the PRINT option is in effect, the program prints the comments.

Usage Note

There must be at least one blank following the asterisk (*) before you enter the text.

END Control Record

The END control record ends ZAP processing. The END record is required and must be the last control record for input from the console.



Usage Notes

1. Before using the ZAP command against MODULE files, you can use the MODMAP command to determine whether a module map exists and what it contains.
2. When a ZAP input file has more than one pair of VER and REP control records, and a VER control record (other than the first) fails, you must remove the records before the failing record and correct the error before you issue the ZAP command again. Otherwise, the file being changed returns to its original status.
3. The REP control record cannot be used to place data in an undefined area such as a Define Storage area. If any part of a data field specified in a pair of VER and REP control records is an undefined area, the system displays warning message DMS248W, and no data replacement occurs. If you do not issue a VER control record prior to the REP control record, some change to data could result. User-defined data can be inserted in undefined areas of text files by using the REP statement described under the LOAD command.
4. If the file to be dumped contains undefined areas (such as a DS or ORG statement in a TXTLIB member), the hexadecimal portion of the dump contains blanks to indicate the corresponding positions are undefined.
5. VER and REP control words can be used to change TXTLIB members produced by FORTRAN compilers that store the length of the compiled text in the END card rather than in the ESD card. However, if a member of this type contains multiple CSECTs, only the first CSECT can be changed by the ZAP program.

6. The TXT records should be in ascending address order. If ZAP finds a TXT record with an address higher than the specified address, it stops scanning for the specified address. This means ZAP control records affect only the data at the first occurrence of the address.
7. When applying ZAPs to a text deck created by a compiler, be aware some compilers, such as FORTRAN, might generate a text deck in which the TXT records are not in ascending address order.
8. If the ZAP command is issued against a file located in an SFS directory, and that file cannot be closed successfully, processing stops with a return code of 31.

Messages and Return Codes

- DMS001E No {*filename*|name *names*] specified [RC=24]
- DMS002E [Output|Overlay] {File[s] |Dataset|Note} [*fn* [*ft*]] not found [RC=20|28|36]
- DMS003E Invalid option *option* (with function *function*) [RC=24]
- DMS007E File *fn ft fm* is not fixed, 80-character records [RC=32]
- DMS014E Invalid {function *function*|keyword *keyword*} [RC=24]
- DMS030E File *fn ft fm* already active [RC=28]
- DMS047E No function specified [RC=24]
- DMS056E File *fn ft fm* contains invalid {name| alias|entry|ESD|RLD} record formats {*entryname*} [RC=32]
- DMS070E Invalid [parameter *parameter*|argument *argument*] [RC=24|28]
- DMS073E Unable to open file]*ddname*|*fn*] [RC=28]
- DMS104S Error *nn* reading file *fn ft fm* [from {disk or directory|XEDIT}] [RC=24]
- DMS190W Invalid control record or NO GO switch set [RC=4]
- DMS191W Patch overlaps; set NO GO switch [RC=4]
- DMS192W Odd number of digits; set NO GO switch [RC=4]
- DMS193W Preceding control record flushed [RC=4]
- DMS194W CSECT not found in [member *membername*|module *module*]; set NO GO switch [RC=4]
- DMS195W Base value invalid; set NO GO switch [RC=4]
- DMS200W Verify reject; set NO GO switch [RC=4]
- DMS208E File *fn ft* is not variable record format [RC=24]
- DMS210E [Library *libname*|File *fn ft*] is on a read-only file mode [RC=36]
- DMS245S Error *nnn* on printer [RC=100]
- DMS246W No loader table present for module *fn*; set NO GO switch [RC=4]
- DMS247W Member membername not found; set NO GO switch [RC=4]
- DMS248W Invalid VER/REP displacement; set NO GO switch [RC=4]
- DMS249I Dummy log entry in file fn ZAPLOG fm
- DMS750I ZAP processing complete
- DMS751I Member *membername* found in library libname
- DMS1137E Object {already|is} locked; deadlock detected} or in use { RC=70|31]

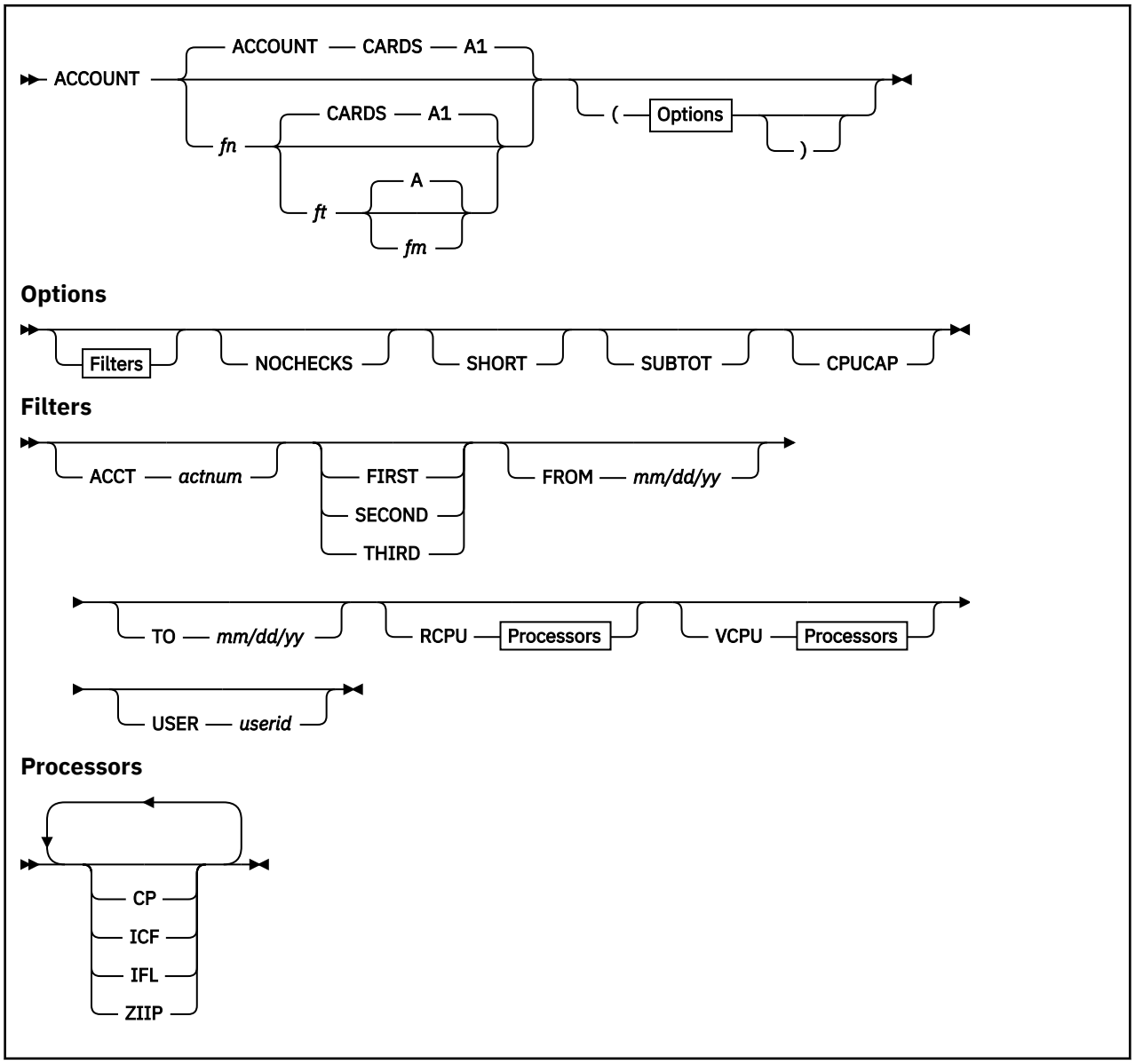
Chapter 3. Utilities

This section contains reference information on the CMS utilities. These utilities are not typically issued by general CMS users, but are mostly used by system programmers. The utilities are listed in alphabetical order.

Each utility description is presented in the following format (some sections might not be included):

- ***Name***: Identifies the name of the utility.
- ***Format***: Shows the syntax of the utility with all of the possible operands and options you can use.
- ***Authorization***: Identifies the type of user who can issue the utility, required CP privilege classes (multiple classes might be required), other authorization requirements, and the location of the utility (if not supplied on the system disk).
- ***Purpose***: States what the utility is used for.
- ***Operands and Options***: Defines the function of each operand and option and any values you can include.
- ***Usage Notes***: Identifies and describes special situations and other considerations that may affect your use of the utility.
- ***Examples***: Provides one or more examples to show how the utility is commonly used.
- ***Responses***: Describes the responses you might receive from the utility on your display device. Responses are normal operational output; they tell you about the execution and effect of the utility. Unlike system messages, utility responses are not prefixed with an identifying number and are not contained in *z/VM: CMS and REXX/VM Messages and Codes*.
- ***Messages and Return Codes***: Lists the messages issued by the utility. Messages not unique to the utility might also be issued. Each message is prefixed with an identifying number. For more information on the messages, including suggested actions, see *z/VM: CMS and REXX/VM Messages and Codes*.

ACCOUNT



Authorization

Systems Programmer

Purpose

Use the ACCOUNT utility to process accounting cards and produce a printed report of the data.

Operands

fn ft fm
specifies the input file to be processed. The default is ACCOUNT CARDS A1.

Options

ACCT *actnum*
limits processing to only the account number specified by *actnum*.

FIRST
creates a report for first shift only (08:30 - 17:15).

SECOND
creates a report for second shift only (17:15 - 24:00).

THIRD
creates a report for third shift only (24:00 - 08:30).

FROM *mm/dd/yy*
sets the date of the earliest data to be processed. Dates before 1972 are not accepted.

TO *mm/dd/yy*
sets the date of the most recent data to be processed.

RCPU
limits processing to only those type 1 accounting cards generated for the specified real CPU type. You can specify more than one real CPU type.

VCPU
limits processing to only those type 1 accounting cards generated for the specified virtual CPU type. You can specify more than one virtual CPU type.

CP
Central Processor

ICF
Internal Coupling Facility

IFL
Integrated Facility for Linux

ZIIP
IBM z Integrated Information Processor

USER *userid*
limits processing to only those accounting cards generated for the specified user ID.

NOCHECKS
inhibits checking to ensure real processor time on each accounting card is not less than the virtual processor time on the card. (If the real processor time is less than the virtual processor time, ACCOUNT considers the accounting card not valid or "bad" and ignores it.) Without this option, ACCOUNT will issue a message each time a bad card is found, and it will print the total number of bad cards on the report.

SHORT
prints only subtotals by "project number". All the accounting information for all users with a certain project number is totaled and these *totals* are printed. Project numbers are defined as the first four characters of the user's account number. Account numbers with the first four characters the same will be grouped together on the report.

SUBTOT
calculates subtotals by "account number". There may be more than one user for each account number; thus, you have subtotals of information for the account numbers and the associated user IDs. Data for each account number is printed on a separate page.

CPUCAP
creates a CPU capability report based on CPU capability accounting records (type D). This option can be combined with all other options. Whenever CPUCAP is combined with FROM, TO, FIRST, SECOND, or THIRD, a CPU capability report is generated just for that time frame. If any other options are also specified, both a regular report and a CPU capability report are generated. ACCOUNT will generate ONE PRT spool file which will contain both reports.

Usage Notes

1. By default, the ACCOUNT utility is located on the system tools disk (MAINT 193).
2. ACCOUNT will only handle one project number per user ID.
3. ACCOUNT will only process cards with accounting record identification codes 01 through 03 and 0D. ACCOUNT will not process cards with other accounting record identification codes.
4. See *z/VM: CP Planning and Administration* for more information about accounting records. For more information about the LOGON and XAUTOLOG commands, see *z/VM: CP Commands and Utilities Reference*.
5. ACCOUNT will generate a PRT spool file.
6. If more than one of the shift options (FIRST, SECOND, and THIRD) are specified, only the one specified last will be in effect because only one can be in effect at a time.
7. If using ACCOUNT to create reports for specific shifts (first, second, or third), you will need to close out the accounting records following each shift. This is accomplished by issuing the following command at 08:30:00 (for third shift), 17:15:00 (for first shift), and at 24:00:00 (for second shift):

   ```
   CP ACNT ALL CLOSE
   ```

8. The account number is specified in the user's directory entry or when a user logs on using ACCOUNT or LOGON or XAUTOLOG.
9. ACCOUNT can only process accounting cards generated in the period of one month at the maximum. Beyond one month, the results cannot be guaranteed to be accurate.
10. When using the RCPU or VCPU option, only the following options will appear in the output: USERID, SESS, CONNECT RATIO, REAL-CPU, and VIRT-CPU. In addition, types 2 and 3 ACCOUNT cards and type 1 ACCOUNT cards for processor types other than what is requested with the RCPU and VCPU options will not be processed. For more information about real CPU (RCPU) and virtual CPU (VCPU) types, see "Setting Up Service Virtual Machines" in *z/VM: CP Planning and Administration* and the "Specialty Engine Support" information in *z/VM: Running Guest Operating Systems*.

Examples

These are some examples of ACCOUNT, using different options.

Entering ACCOUNT with no options, the results are:

```
1VM SYSTEM USAGE OVER THE PERIOD 02/18/01 TO 02/18/01     ALL SHIFTS      USER  ALL         ACCT  ALL
USERID    SESS   CONNECT RATIO   REAL-CPU   VIRT-CPU   PG READ  PG WRITE       SIO    PUNCH    PRINT   READ CYL/BLK TDSK TAPE  DISK
OPERATOR     2 000000:02 ***** 0000:00:00 0000:00:00      1523      1680       704        0      166      0       3    2    0    0
OPEREREP     1 000000:04 ***** 0000:00:00 0000:00:00         0       938       352        0        0      0       0    0    0    0
OPERSYMP     1 000000:04 ***** 0000:00:00 0000:00:00         0         1       352        0        0      0       0    0    0    0
TOTALS       4 000000:11 ***** 0000:00:00 0000:00:00      1523      2619      1408        0      166      0       3    2    0    0
```

Entering ACCOUNT with the SUBTOT option, the results are:

```
1VM SYSTEM USAGE OVER THE PERIOD          01/18/01 TO 01/18/01     ALL SHIFTS      USER  ALL         ACCT ACCNT2
USERID    SESS   CONNECT RATIO   REAL-CPU   VIRT-CPU   PG READ  PG WRITE       SIO    PUNCH    PRINT     READ CYL/BLK TDSK TAPE DISK
MYIDFOUR     1 000000:00 ***** 0000:00:00 0000:00:00         1         4       352        0       45       0       0    0    0    0
MYIDTHRE     0 000000:00 ***** 0000:00:00 0000:00:00         0         0         0        0        0       0       2    1    0    0
PROJ TOT     1 000000:00 ***** 0000:00:00 0000:00:00         1         4       352        0       45       0       2    1    0    0

1VM SYSTEM USAGE OVER THE PERIOD          01/18/01 TO 01/18/01     ALL SHIFTS      USER  ALL         ACCT ACCNT3
USERID    SESS   CONNECT RATIO   REAL-CPU   VIRT-CPU   PG READ  PG WRITE       SIO    PUNCH    PRINT     READ CYL/BLK TDSK TAPE DISK
MYIDFIVE     1 000000:04 ***** 0000:00:00 0000:00:00         2       938       352        0        0       0       0    0    0    0
MYIDSIX      1 000000:04 ***** 0000:00:00 0000:00:00         3         1       352        0        0       0       0    0    0    0
PROJ TOT     2 000000:09 ***** 0000:00:00 0000:00:00         5       939       704        0        0       0       0    0    0    0

1VM SYSTEM USAGE OVER THE PERIOD          01/18/01 TO 01/18/01     ALL SHIFTS      USER  ALL         ACCT  TOTALS
USERID    SESS   CONNECT RATIO   REAL-CPU   VIRT-CPU   PG READ  PG WRITE       SIO    PUNCH    PRINT     READ CYL/BLK TDSK TAPE DISK
 TOTALS      3 000000:09 ***** 0000:00:00 0000:00:00         6       943      1056        0       45       0       2    1    0    0
```

Entering ACCOUNT with the SHORT option, the results are:

```
1VM SYSTEM USAGE OVER THE PERIOD 02/18/01 TO 02/18/01     ALL SHIFTS      USER  ALL        ACCT  ALL
PROJNO    SESS   CONNECT RATIO   REAL-CPU   VIRT-CPU   PG READ  PG WRITE       SIO    PUNCH    PRINT   READ CYL/BLK TDSK TAPE  DISK
2NDL         4 000000:11 ***** 0000:00:00 0000:00:00      1523      2619      1408        0      166      0       3    2    0    0
TOTALS       4 000000:11 ***** 0000:00:00 0000:00:00      1523      2619      1408        0      166      0       3    2    0    0
```

Entering ACCOUNT with either the RCPU or VCPU option, the results are:

```
1VM SYSTEM USAGE OVER THE PERIOD 02/09/07 TO 02/09/07     ALL SHIFTS      USER  ALL       ACCT  ALL
USERID    SESS   CONNECT RATIO   REAL-CPU   VIRT-CPU
MIXENG3     35 000008:45 ***** 0000:00:00 0000:00:00
MIXENG4      5 000004:07 14822 0000:00:01 0000:00:01
SCHNEIDE     5 000006:30 23424 0000:00:01 0000:00:00
WPMURPHY     1 000005:16 03163 0000:00:06 0000:00:05
 TOTALS     46 000024:38 11091 0000:00:08 0000:00:06
```

Messages and Return Codes

- DMS029E Invalid parameter *parameter* in the option option field [RC=24]
- DMS104S Error *nnn* reading file fn ft fm from disk or directory [RC=1]
- DMS123S Error *nn* in printing file *fn ft fm* [RC=1 xxx]
- DMS161E Invalid free storage release call from *addr*, error code nn [RC=0]
- DMS2163W Unable to get space – will ignore new userids.
- DMS2164W Bad time on card:
- DMS2165E Invalid {month|day|year} in FROM/TO option. [RC=3|4|5]
- DMS2260E Specified USER ID or account number not found in input file. [RC=6]
- DMS2261E Bad cards skipped:
- DMS2262E Users not active during period:
- DMS2263E No records for virtual machine resource usage, dedicated devices, and temporary disk space meet the specified criteria.
- DMS2263E No records meet the specified criteria. [RC=7]
- DMS2264E The FROM date is later than the TO date [RC=8]
- DMS2265E No CPU capability records found.
- DMS2265E No CPU capability records found during specified period.
- DMS2266I CPU capability records encountered, use CPUCAP option for CPU capability report.

Additional system messages might be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

Return codes:

RC
Meaning

0
Operation complete—no errors

1
Input file error—FSOPEN or FSREAD error

3
Invalid month in FROM/TO option

4
Invalid day in FROM/TO option

5
Invalid year in FROM/TO option

6
Operand for USER or ACCT option was not found in input file

7
No cards meet the specified criteria

8
The FROM date is greater than the TO date

24
Operand for USER, ACCT, RCPU, or VCPU is missing or incorrect

100
Explanation complete (when '?' specified)

1xxx
Return code *xxx* and error *nn* from the PRINTL macro or the CP CLOSE command

AUDITOR

Authorization

Systems Programmer

Purpose

Use the AUDITOR utility to monitor service virtual machines (SVMs) running on your system. AUDITOR tells you which SVMs are running properly, are logged off, are in a disabled wait state, or have failed tests made by test routines you create. AUDITOR not only checks SVMs to see if they are running properly, it can restart a service virtual machine that is disabled or logged off.

AUDITOR is most useful as an automated tool that runs unattended from a dedicated virtual machine. As such, AUDITOR is started from the PROFILE EXEC in the AUDITOR virtual machine, which is then disconnected. Be sure to uncomment the call to the AUDITOR EXEC in the AUDITOR's PROFILE EXEC when using AUDITOR for the first time. (See “Beginning SVM Monitoring” on page 1308 for specific information on this kind of startup.)

AUDITOR provides two files that drive its processing: AUDITOR OPTIONS and AUDITOR CONTROL. Before running AUDITOR, you tailor these files to suit your own installation. For information on the virtual machine requirements for running AUDITOR, and details on tailoring the AUDITOR input files, see “AUDITOR: Requirements, Setup, and Use” on page 1302.

Once AUDITOR is running, authorized users can issue AUDITOR subcommands from their own virtual machine. These subcommands give the status of the SVMs, restart AUDITOR, and stop AUDITOR.

To send AUDITOR subcommands from a user ID on the same system, use the MSG (or SMSG) or TELL command:

```
MSG userid  subcommand
TELL userid  subcommand
```

where *userid* is the virtual machine running AUDITOR, and *subcommand* is the AUDITOR subcommand. To send AUDITOR subcommands from a user ID on another system, use RSCS and the SMSG or TELL command:

```
SMSG net_nodeid MSG nodeid  userid  subcommand
TELL userid at nodeid subcommand
```

where *net_nodeid* is the RSCS virtual machine the local node communicates with, *nodeid* is the local node for AUDITOR, *userid* is the virtual machine running AUDITOR, and *subcommand* is the AUDITOR subcommand.

AUDITOR will respond by returning any pertinent information to your console.

Subcommands

The STATE subcommand returns the status of the monitored SVMs. STATE returns the following status values to your console:

UP
The SVM is running normally.

DOWN
The SVM is not logged on, or is in a disabled wait state.

PD
The last time AUDITOR checked, the SVM was okay; this time, however, it failed a test made by your test routine. This routine is identified by the *test_exit* value (see Table 62 on page 1305) for the tested SVM, as specified in the AUDITOR CONTROL file.

OFF
The SVM exceeded its *max_errors* value (see Table 62 on page 1305). In this case, AUDITOR keeps checking the SVM, but no longer reports its status. Nor does AUDITOR try to restart the SVM, even though its force and autolog flag may be set in the AUDITOR OPTIONS file. Someone must actually intervene to fix and restart the SVM. Then, the next time AUDITOR checks the SVM, AUDITOR will detect the machine state change, reset its counters, and reset the SVM status to UP.

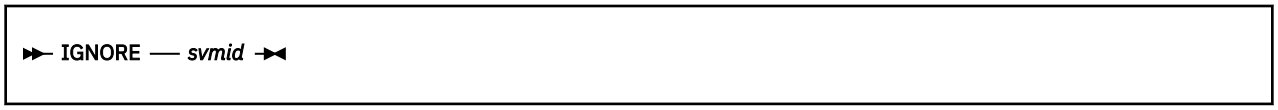
SERVICE
Someone is logged on to the SVM's terminal. AUDITOR assumes the SVM is being serviced and will not attempt to log it off, even though its force and autolog flag may be set in the AUDITOR OPTIONS file.

RECYCLED
The SVM cycled three or more times since midnight, or since AUDITOR last recycled, whichever occurred later.

CYCLE
AUDITOR found the SVM in a DOWN state, and successfully logged it back on.

FAILURE
The last time it checked, AUDITOR found the SVM in a PD state (that is, the SVM failed a test made by your test routine), and it is still failing.

IGNORE
The SVM is not being tested.



The IGNORE subcommand halts AUDITOR's checking and restarting of SVM *svmid* (where *svmid* is the user ID of the SVM), without affecting its checking and restarting of other SVMs. The IGNORE and RESET subcommands work together, switching the checking and restarting of the SVM off and on.

Note: For a user ID to issue this command, it must be the user ID of the specified SVM or a user ID authorized in the AUDITOR OPTIONS file.

The RESET subcommand instructs AUDITOR to resume checking and restarting of SVM *svmid* (where *svmid* is the user ID of the SVM), currently in an OFF or IGNORE state. When the SVM is reset, its error counts are reset, and the SVM is checked again at its regularly scheduled time. The RESET and IGNORE subcommands work together, switching the checking and restarting of the SVM on and off.

Note: For a user ID to issue this command, it must be the user ID of the specified SVM or a user ID authorized in the AUDITOR OPTIONS file.

The RESTART subcommand causes the running AUDITOR machine to begin its initialization again. This subcommand can be used in conjunction with a restart exit exec to perform such tasks as prompting

AUDITOR to reread the AUDITOR CONTROL and AUDITOR OPTIONS files, which may have been updated. Items such as SVM error counting and SVM monitoring that are set at initialization will be reset.

The STOP subcommand shuts AUDITOR down.

The CP subcommand issues a CP command. You can issue this command only from the console of the virtual machine running AUDITOR.

The CMS subcommand issues a CMS command. You may only issue this command from the console of the virtual machine running AUDITOR.

The HELP subcommand returns general help information about all AUDITOR subcommands. If you type a specific AUDITOR subcommand after 'HELP,' only information about that subcommand is returned.

Usage Notes

1. By default, the AUDITOR utility is located on the system tools disk (MAINT 193).
2. The virtual machine running the AUDITOR utility requires CP privilege classes A and B, C or E, and G. To issue AUDITOR subcommands, a user must be authorized in the AUDITOR OPTIONS file.

Messages and Return Codes

- DMS002E File *fn* not found [RC=28]
- DMS023W No extension specified for filename *fn* [RC=1]
- DMS2300I {*userid*|AUDITOR} running on userid *userid* at *nodeid*.
- DMS2301S Insufficient privilege class for command: *command*. [RC=1]
- DMS2302S There is insufficient storage to run {*userid*|AUDITOR}. [RC=1]
- DMS2304S A-disk or directory must be R/W to run {*userid*|AUDITOR}. [RC=1]
- DMS2305S A-disk or directory must be less than *num*% full to run {*userid*|AUDITOR}. [RC=1]
- DMS2306S Invalid entry in line *num* of {*fn ft*|AUDITOR CONTROL}: Invalid {test period|maximum error value|flags} for SVM *userid*. [RC=12]
- DMS2306S No valid entries in the {*fn ft*|AUDITOR CONTROL} file: No SVM's to monitor [RC=12]
- DMS2308S Error in line *num* of {*fn ft*|AUDITOR CONTROL}: *execname* EXEC not found.
- DMS2308S Error in line *num* of {*fn ft*|AUDITOR OPTIONS}: Invalid DISKMAX value *value*.
- DMS2308S Error in line *num* of {*fn ft*|AUDITOR OPTIONS}: Invalid exit type *type*.
- DMS2308S Error in line *num* of {*fn ft*|AUDITOR OPTIONS}: Invalid record type *type*.
- DMS2308S Error in line *num* of {*fn ft*|AUDITOR OPTIONS}: Invalid reset time.
- DMS2308S Error in line *num* of {*fn ft*|AUDITOR OPTIONS}: No userid specified for *keyword* keyword.

- DMS2309I *Userid* is being serviced at *time*.
- DMS2310I Next SVM to be tested is *userid* at *nodeid* in *num* seconds.
- DMS2311S Unexpected return code *rc* from LOCATE. [RC=8]
- DMS2312W Error condition detected by test exec in SVM *userid* at *nodeid*.
- DMS2313W Unexpected return code *rc* from *execname* for SVM *time*.
- DMS2314I *userid* is now logged on at *time*.
- DMS2315I *userid* is not logged on *nodeid*.
- DMS2315W *userid* is not logged on at *time*.
- DMS2316I SVM *userid* was not logged on *nodeid*. It has been restarted.
- DMS2317I SVM *userid* is in disabled wait state at *time*.
- DMS2318S Unexpected return code *rc* from *execname* exec. Unable to autolog SVM *userid*.
- DMS2319I SVM *userid* stopped running on *nodeid*. It was logged off the system and restarted.
- DMS2320W SVM *userid* has stopped running on *nodeid*. Please check it.
- DMS2321E Unknown user command: *command*. No action taken.
- DMS2322E {CMS|CP} command only valid from the console.
- DMS2323I Restarting {*userid*|AUDITOR} at *nodeid*.
- DMS2324I Shutting down *userid* at *nodeid*.
- DMS2325I Shutdown of *userid* completed. *hh:mm:ss*.
- DMS2326S Unexpected return code *rc* from *command*. [RC=1*xxx*]
- DMS2327I *Fileid* created by *userid* at *nodeid* (VM).
- DMS2328I Processing command *command* for SVM *userid*.
- DMS2329S An autolog exit is required to autolog service virtual machines. A record type "EXIT AUTOLOG" needs to be in the {*fn ft*|AUDITOR OPTIONS} file. [RC=12]
- DMS2330W Unexpected return code *rc* from EXECIO. *userid* will no longer write to its journal.
- DMS2331W Unexpected return code *rc* in NOTE EXEC. *userid* will not send notes.
- DMS2332W Unexpected return code *rc* in TELL EXEC to *userid*
- DMS2333W AUDITOR could not deliver a message because: [*reason*]
- DMS2334W There was an unexpected return code from the note exec.
- DMS2335W There was an unexpected return code from the tell exec.
- DMS2352I Unknown command. Issue HELP with no arguments for a list of valid commands.
- DMS2353W Unexpected return code *rc* from COPYFILE. {*userid*|AUDITOR} will no longer write to its journal.
- DMS2354I Service machine '*userid*' has been dropped from testing.
- DMS2355I Service machine '*userid*' has been reset and testing will resume.
- DMS2356I Service machine '*userid*' is not supported by {*userid*|AUDITOR}.
- DMS2357I You must specify a machine id for the '*command*' command. [RC=24]
- DMS2358S EXECIO error while reading file *fileid*. EXECIO RC = *rc*. [RC=24]
- DMS2359W Unexpected return code *rc* from exit *exec_name*.
- DMS2359W Unexpected return code *rc* from exit *exec_name*. The exit has been disabled.
- DMS2360S Undefined Variable on line *num* of *exec_name*. [RC=9999]
- DMS2360S Undefined Variable on line *num* of *exec_name*: *record* [RC=9999]

Return codes:

RC
Meaning

0
Normal exit

1
AUDITOR machine does not meet requirements to run AUDITOR

8
Error in LOCATE command

12
Invalid entry in an AUDITOR input file

24
EXECIO error

28
Required file not found

1xx
Return code *xx* from WAKEUP utility

2xx
Return code *xx* from XMITMSG or an exit exec

9999
Fatal error

AUDITOR: Requirements, Setup, and Use

The AUDITOR utility monitors service virtual machines (SVMs) running on your system. The AUDITOR utility is usually set up to run unattended on a dedicated virtual machine.

The virtual machine running AUDITOR requires:

- CP privilege classes A and B, C or E, and G, or the proper privilege classes to issue the FORCE, XAUTOLOG, LOCATE, DISPLAY H, and MSGNOH commands.
- 5 MB of virtual storage.
- A R/W disk or SFS directory with sufficient space to store AUDITOR's work and output files. About 5 cylinders of 3380 DASD, or the equivalent for your hardware, will be required in support of AUDITOR.

AUDITOR provides two files that drive its processing: AUDITOR OPTIONS and AUDITOR CONTROL. Before running AUDITOR, you tailor these files to suit your own installation.

AUDITOR OPTIONS File

The AUDITOR OPTIONS file describes how AUDITOR will run at your installation. You use it to specify AUDITOR's administrator, authorized users, and exit routines. Exit routines are routines you create to log on SVMs, and to process any reader files sent to the virtual machine running AUDITOR. The AUDITOR OPTIONS file also specifies how full AUDITOR's A-disk or directory can get before AUDITOR stops processing, and a time for AUDITOR to restart its error counts for all SVMs.

Statements in the AUDITOR OPTIONS file consist of an option and a value or values. For example,

```
AUTH OPERATOR *
```

is an option statement telling AUDITOR to give user authority to the OPERATOR user ID, when OPERATOR is at the same node as the SVM running AUDITOR.

The following table describes the ADMIN, AUTH, DISKMAX, and RESETTIME options, and their values and defaults.

Table 60. AUDITOR OPTIONS Options ADMIN, AUTH, DISKMAX, and RESETTIME

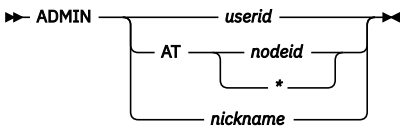
OPTION	Description
	Specifies the user ID of the current AUDITOR administrator. In the event of a severe AUDITOR error, the AUDITOR administrator will receive a message and a reader file noting the error. Specify the administrator as `ADMIN` *userid* for a local user ID, or as `ADMIN` *userid* `AT` *nodeid* for a remote user ID. Alternatively, you can specify a nickname, as defined in the NAMES file on the virtual machine running AUDITOR: `ADMIN` *nickname* **The default is OP**. OP searches for and forwards information to the user ID running the Programmable Operator Facility. This user ID serves as the AUDITOR administrator in the distributed environment, where AUDITOR is most commonly used. If your application runs in the departmental (local) environment, specify a user ID such as ADMIN or MAINT for the AUDITOR administrator.
`AUTH` *userid nodeid*	Specifies the user ID and node ID of an authorized AUDITOR user. You can give user authority to more than one user ID, but only to one user ID in each AUTH statement. Specify an asterisk (*) for *nodeid*, to indicate an authorized user ID at the same node as the virtual machine running AUDITOR: `AUTH` *userid* `*`

Table 60. AUDITOR OPTIONS Options ADMIN, AUTH, DISKMAX, and RESETTIME (continued)

OPTION	Description
DISKMAX *percent*	Specifies the highest percentage of used A-disk or SFS directory space AUDITOR will allow before it ends processing and stops. The default is 90.
RESETTIME *hh:mm:ss*	Specifies the time of day AUDITOR will restart the error counts for the *max_errors* values set in the AUDITOR CONTROL file. At this time AUDITOR rechecks all SVMs, including the SVMs AUDITOR stopped checking because they had exceeded their *max_errors* values. Specify the reset time in *hh* (00-24) hours, *mm* minutes, and *ss* seconds. The default is 00:00:00 (midnight).

The following table describes the EXIT option, its values and defaults.

PI

Table 61. AUDITOR OPTIONS Option EXIT

OPTION	Description
EXIT *type execname*	Specifies an exit routine—an exec—which you can create to extend AUDITOR's function. Only one of each type of exit is allowed. AUDITOR uses five *types* of execs: **AUTOLOG** An exec for handling your SVM logons. The exec specified for AUTOLOG is passed the SVM ID name as an argument. **RDR** An exec for handling reader files sent to AUDITOR. No arguments are passed to the exec specified for RDR. **SENDSTAT** An exec for sending SVM data from a monitored system to the central site. The exec used with SENDSTAT is executed anytime an SVM's status is checked. The exec specified for SENDSTAT is passed *userid* and *state* as arguments. **If you are working with centrally managed systems**, use SENDSTAT together with the SYSWATCH utility. (See "Using AUDITOR with SYSWATCH" on page 1379.) **NEWDAY** An exec for doing processing tasks on a day-to-day basis—tasks like daily console and log cleanup. No arguments are passed to the exec specified by NEWDAY. **RESTART** An exec for performing actions such as reaccessing disks on IPL or recycle, without having to shut AUDITOR down. No arguments are passed to the exec by RESTART. For example, you could write an exit routine to reaccess the disk containing the AUDITOR CONTROL and AUDITOR OPTIONS files. Then, when updates of these files are sent to the system from the central site you can issue the RESTART subcommand to load the updates without having to shut AUDITOR down.

Table 61. AUDITOR OPTIONS Option EXIT (continued)

OPTION	Description

EXIT *type execname* (continued)	Here are some exit examples:
	
	To indicate a logon (autolog) exec, specify *type* as AUTOLOG, and *execname* as the file name of your logon exec:
	
	`EXIT AUTOLOG` *execname*
	
	(Use the sample logon exec provided, AUDALOG EXEC, to log on SVMs. If you decide to use AUDALOG, specify
	
	`EXIT AUTOLOG AUDALOG`
	
	as your EXIT option.)
	
	To forward SVM status information from a monitored system to a central site, specify *type* as SENDSTAT, and *execname* as the file name of your forwarding exec:
	
	`EXIT SENDSTAT` *execname*
	
	Use the sample exit routine provided, SENDSTAT EXEC, to use with this exit if you also use the SYSWATCH utility. The SENDSTAT EXEC gets the status information, formats it for SYSWATCH, and sends it to the central monitoring system named in SYSWATCH CONTROL. If you decide to use SENDSTAT, specify
	
	`EXIT SENDSTAT SENDSTAT`
	
	as your EXIT option. (See "Using AUDITOR with SYSWATCH" on page 1379 for details on how to use SENDSTAT with SYSWATCH.)
	
	To indicate a reader exec, specify *type* as RDR, and *execname* as the file name of your reader exec:
	
	`EXIT RDR` *execname*
	
	Use of the EXIT option yields one of two return codes:
	
	RC
	Message or Meaning
	
	0
	Normal Exit
	
	x
	Unexpected return code x
	
	If a test on an SVM yields a return code of 1, AUDITOR will increase that SVM's error count by one.
	
	If choosing to implement a RDR exit, the exit should minimally handle reader files in one of the following ways:
	
	• Place the reader files on HOLD
	• Receive the reader files
	• Purge the reader files
	• Change the class of the reader files to a class different from the class of the reader
	
	If the reader files are not handled, AUDITOR may appear to loop or hang when testing SVMs.
	
	Use of one or all of these exec types is optional—however, if you do not specify a logon exec, AUDITOR cannot autolog your SVMs. If you do not specify a reader exec to handle files sent to AUDITOR's virtual machine, these files will remain in the AUDITOR machine's reader. AUDITOR itself does no spool file handling.

Execs Used by the AUDITOR OPTIONS File

Exit routines are your own execs, which you specify using the EXIT option described in Table 61 on page 1303. AUDITOR requires you write or provide it with exit routines, if you want to process reader files sent to AUDITOR's virtual machine, or log on SVMs.

Because a likely use of AUDITOR is to log on SVMs, a logon routine called AUDALOG EXEC is provided, along with the AUDITOR OPTIONS file. You can use AUDALOG EXEC instead of creating a logon exec. AUDALOG EXEC opens a file named AUDALOG SVMLIST, which contains a table of SVMs that AUDITOR reads to log on the SVMs. So, if you use AUDALOG EXEC for automatic logons, be sure to modify the AUDALOG SVMLIST table with a list of SVMs you will allow AUDITOR to log on.

If you are working with centrally managed systems and you want to send SVM status information to your central site, a routine is provided to do so. The routine is called SENDSTAT EXEC. See "Using AUDITOR with SYSWATCH" on page 1379 for details on how to use SENDSTAT EXEC with SYSWATCH.

PI end

Sample AUDITOR OPTIONS File

Each option statement in the AUDITOR OPTIONS file consists of an option followed by a value. Blank lines and lines that begin with an asterisk (*) are ignored.

A sample AUDITOR OPTIONS file follows; use it as a guide when tailoring the AUDITOR OPTIONS file provided. Only some of the exits in this options file are shipped as samples.

```
************************************************************************
* AUDITOR OPTIONS
************************************************************************

************************************************************************
*  AUDITOR Options File                                                *
*  Record Types:  ADMIN userid (at nodeid)  - AUDITOR Administrator   *
*                 AUTH userid nodeid  - Authorized User               *
*                 EXIT type execname  - User Exit                     *
*                 DISKMAX n  - Maximum A-Disk percent full            *
************************************************************************
* Assign OP AT CENTRAL as the AUDITOR administrator user ID
ADMIN OP AT CENTRAL
*
* Assign some local user IDs as authorized AUDITOR users
AUTH  OPERATOR *
AUTH  MAINT    *
AUTH  ADMIN    *
*
* Also assign some user IDs at node CENTRAL as authorized AUDITOR users
AUTH  ADMIN    CENTRAL
AUTH  USER1    CENTRAL
*
* Use the AUDALOG exec for logging on SVMs
EXIT  AUTOLOG  AUDALOG
*
* Use a locally-written exec, RDRFILES, for handling AUDITOR's
* reader files
* EXIT  RDR  RDRFILES
*
* Use a locally-written exec, NEWDAY, for handling daily console and
* log cleanup
* EXIT NEWDAY NEWDAY
*
* Use a locally-written exec, INITIAL, for reaccessing disks on IPL and
* recycle
* EXIT RESTART INITIAL
*
* Use the SENDSTAT exec for forwarding SVM status to the central site
EXIT SENDSTAT SENDSTAT
*
* Tell AUDITOR to stop running when its A-disk or directory is 85% full
DISKMAX 85
*
* Tell AUDITOR to reset all SVM error counters, including those
* for SVMs that have exceeded their max_errors value
RESETTIME 01:00:00
```

Figure 72. Sample AUDITOR OPTIONS File

AUDITOR CONTROL File

The AUDITOR CONTROL file contains a series of control statements. Each control statement describes one service virtual machine you want to monitor, and assigns to it SVM data for all of the values in Table 62 on page 1305. A control statement must present this data in the following order:

Table 62. AUDITOR CONTROL Values

VALUE	Description
svmid	Specifies the user ID of a service virtual machine (the SVM ID) AUDITOR will monitor.

Table 62. AUDITOR CONTROL Values (continued)	
VALUE	**Description**
test_interval	Specifies how often the SVM will be checked. Specify the test interval in the format *hh:mm:ss*, where *hh* is hours, *mm* is minutes, and *ss* is seconds. For example, the test interval value `00:05:00` instructs AUDITOR to check the SVM every 5 minutes.
autolog_flag	Specifies whether AUDITOR should log on the SVM if it is not logged on. Specify *1* to tell AUDITOR to autolog (log on) the SVM if it is logged off; specify *0* to leave it alone. If you specify *1*, you must additionally specify an AUTOLOG exit routine in your AUDITOR OPTIONS file.
force_&_autolog_flag	Specifies whether AUDITOR should log off and then log on the SVM if it is in a disabled wait state. Specify *1* to tell AUDITOR to force (log off) and then autolog (log on) the SVM if it is in a disabled wait state; specify *0* to leave it alone. If you specify *1*, you must additionally specify an AUTOLOG exit routine in your AUDITOR OPTIONS file.
PI	
test_exit	Specifies the name of an exec you have written to test the SVM. Use a test exec if you need to test an SVM for conditions AUDITOR itself does not test for. For example, you might write a test exec to test if an SVM is actively processing or sitting idle. A test exec runs from AUDITOR's virtual machine. AUDITOR invokes the test exec but passes no parameters to it. A test exec should return one of three return codes: **RC** **Message or Meaning** **0** Normal exit **1** There is a problem with the SVM **x** Unexpected return code x from test exec If a test on an SVM yields a return code of 1, AUDITOR will increase that SVM's error count by one. Specify the file name of the test exec, if you have written one; otherwise, specify *NONE*. A test exit will only be called when the SVM it tests is up and running fine. These test exits will not be executed if the SVM has to be autologged or forced and auto logged. Use of a test exec is optional.
PI end	
max_errors	Specifies the maximum number of consecutive errors AUDITOR will allow when checking the SVM. After the maximum is met, AUDITOR will stop checking the SVM. AUDITOR counts as an error any instance where it: 1) checks the SVM and finds it logged off or in a disabled wait state, or 2) receives a return code of 1 from a test_exit testing the SVM. The error count is reset at the time specified in the AUDITOR OPTIONS file.
notify_user	Specifies the user ID to be notified if there is a problem with the SVM. Specify the user as *userid* for a local user ID, or as *userid* AT *node* for a remote user ID. Alternatively, a nickname defined on the virtual machine running AUDITOR can be specified, instead of a user ID.

Sample AUDITOR CONTROL File

Here is an example of a control statement you might find in an AUDITOR CONTROL file:

```
PVM  00:01:30  1  1  NONE  10  MAINT AT CENTRAL
```

This control statement tells AUDITOR to:

1. Monitor the service virtual machine whose user ID is PVM (`PVM`).
2. Check PVM at test intervals of a minute and a half (`00:01:30`).
3. Log on PVM if it is logged off (`1`).
4. Log off PVM and then log it back on, whenever it is in a disabled wait state (`1`).
5. Specify no additional exit routines shall test PVM (`NONE`) if steps 3 and 4 did not have to be done.
6. Stop checking PVM after 10 errors (`10`).
7. Notify the MAINT user ID at node CENTRAL of any problems (`MAINT AT CENTRAL`).

When preparing your CONTROL file, use one control statement to describe each SVM AUDITOR checks. In every control statement, be sure to specify all the Table 62 on page 1305 values, in the order they appear in the table. Blank lines and lines that begin with an asterisk (*) are ignored.

A sample AUDITOR CONTROL file follows; use it as a guide when tailoring the AUDITOR CONTROL file provided:

```
*************************************************************************
* AUDITOR CONTROL
*************************************************************************

*************************************************************************
* MACHINE     TEST    AUTO FORCE TEST     MAX   NOTIFY
*   ID      INTERVAL  LOG  &AUTO EXIT     ERRS  USER ID
*************************************************************************
 CMSAPPL    00:01:00   1     1   NONE      10   ADMIN
 CMSSERV    00:30:00   1     1   NONE      10   ADMIN
 CMSSFS     00:01:00   1     1   NONE      10   ADMIN
```

Figure 73. Sample AUDITOR CONTROL File

Output from AUDITOR

As AUDITOR goes about its checking and testing process, it sends:

- Severe error messages about AUDITOR to the AUDITOR administrator
- Messages and problem notices about the SVMs to the users who maintain the SVMs
- Processing information to its virtual console
- Subcommand responses to the console of the authorized user who issued the subcommand

In addition, AUDITOR logs information about its processing in an ongoing journal file, and notifies specified user IDs about severe errors by sending them reader files.

AUDITOR JOURNAL File

AUDITOR records its progress and error messages, and its processing of subcommands from authorized users, in a file named AUDITOR JOURNAL. AUDITOR creates this journal file automatically. As it logs each activity, AUDITOR notes the date and time in the file.

AUDITOR continuously updates its journal file, which resides on the A-disk or directory of AUDITOR's virtual machine. Because the journal file can become quite large, AUDITOR monitors the size of the file to make sure it never exceeds 4000 records. When this maximum file size is reached, the first 1000 records are deleted to make room for the newest ones.

A sample AUDITOR JOURNAL file follows:

```
*************************************************************************
*                                                                       *
*                     Sample AUDITOR JOURNAL File                       *
*                                                                       *
*                                                                       *
*************************************************************************
01/04/28 14:49:27 Shutdown of AUDITOR completed.
01/04/28 14:49:56 AUDITOR running on user ID AUDITOR at CENTRAL.
01/04/28 14:52:05 CMSSERV is not logged on at 14:52:04.
01/04/28 14:52:38 SVM CMSSERV was not logged on CENTRAL. It has been restarted.
01/04/28 14:53:08 CMSSERV is now logged on 14:53:08.
01/04/28 14:53:25 Processing command STOP for SVM AUDITOR AT CENTRAL.
01/04/28 14:53:25 Shutting down AUDITOR.
01/04/28 14:53:25 Shutdown of AUDITOR completed.
01/04/28 15:13:57 AUDITOR running on user ID AUDITOR at CENTRAL.
01/04/28 15:15:30 CMSSERV is not logged on at 15:15:30.
01/04/28 15:16:20 SVM CMSSERV was not logged on CENTRAL. It has been restarted.
01/04/28 15:16:29 Processing command STATE for SVM AUDITOR AT CENTRAL.
01/04/28 15:16:50 CMSSERV is now logged on 15:16:50.
01/04/28 15:17:04 Processing command CMS Q DISK for SVM AUDITOR AT CENTRAL.
01/04/28 15:17:20 Processing command STOP for SVM AUDITOR AT CENTRAL.
01/04/28 15:17:20 Shutting down AUDITOR.
01/04/28 15:17:20 Shutdown of AUDITOR completed.
```

Figure 74. Sample AUDITOR JOURNAL File

Reader Files

AUDITOR sends a console message about each severe error to the AUDITOR administrator (for AUDITOR problems), and to the user who maintains the SVM (for SVM problems). In addition, AUDITOR sends them reader files describing these errors. Sending severe error messages to both the console and the reader ensures these important messages will reach their specified user IDs, even if the user IDs are logged off when the message is sent.

Beginning SVM Monitoring

To use the AUDITOR utility, proceed as follows:

1. Tailor the AUDALOG EXEC logon routine for use at your installation, or create your own autolog exec for logging on SVMs. In addition, you may want to create an exec for handling reader files received by AUDITOR. These execs are the user exit routines you will specify in your AUDITOR OPTIONS file.
2. Tailor the AUDITOR OPTIONS file for use at your installation. **If you are working with centrally managed systems** and some of them have unique SVMs, create node control files for these systems.
3. Create your own test execs, if desired. These execs are the test exits you will specify in your AUDITOR CONTROL file.
4. XEDIT the AUDITOR PROFILE EXEC and uncomment the call to the AUDITOR EXEC. Change:

   ```
   /*'EXEC AUDITOR'*/
   to
   'EXEC AUDITOR'
   ```

5. Tailor the AUDITOR CONTROL file for use at your installation.
6. Start AUDITOR and test your input files and execs. To start AUDITOR, log on to the AUDITOR virtual machine and enter the command:

   ```
   profile
   ```

   Next, enter the STATE subcommand:

   ```
   state
   ```

   Once you have verified your input files and execs work correctly, enter the command

   ```
   #cp disc
   ```

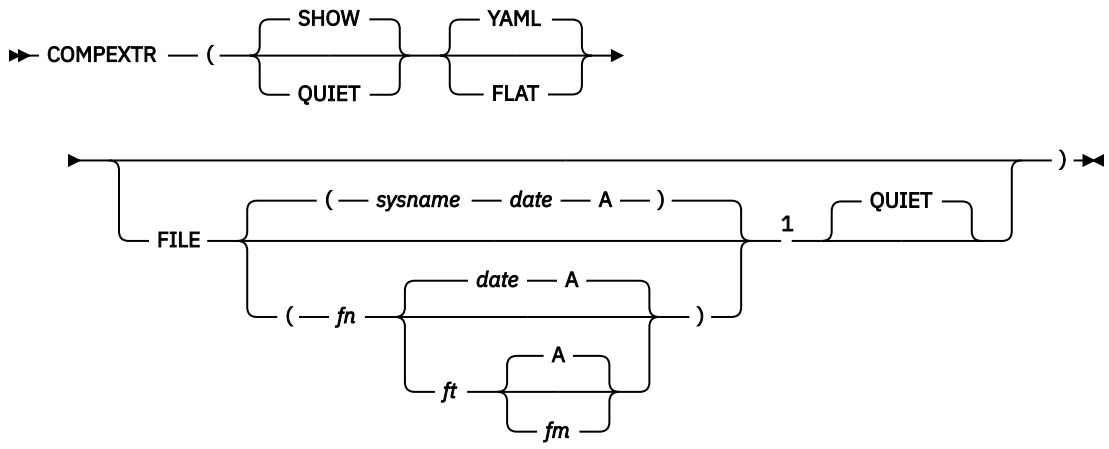
   to disconnect your terminal from the system. The AUDITOR virtual machine will continue running AUDITOR.

Invoking AUDITOR Automatically with AUTOLOG1

If you want AUDITOR to start automatically whenever the system is started, add an entry for the virtual machine running AUDITOR to AUTOLOG1's PROFILE EXEC. See *z/VM: CP Planning and Administration* for information on how to use AUTOLOG1.

When you have completed the steps in this section, the AUDITOR virtual machine will be running disconnected, with AUDITOR monitoring your SVMs. In the future, however, AUTOLOG1 will start AUDITOR for you. Whenever you IPL the system, AUTOLOG1 will log on the AUDITOR virtual machine, and AUDITOR will begin SVM monitoring automatically.

COMPEXTR



Notes:

[1] The FILE option implicitly suppresses console output unless SHOW is specified.

Authorization

System Administrator

Purpose

Use the COMPEXTR command to poll the z/VM system for security-relevant data.

Options

QUIET
suppresses console output.

SHOW
shows console output, even if it was automatically suppressed by the FILE option.

FILE [([*fn* [*ft* [*fm*]]] [)]]
creates an output file. The default filename is the CP system name as shown by `CP Q USERID`. The default filetype is the date in the format *yyyymmdd* (four-digit year, two-digit month, and two-digit day-of-month). Console output is suppressed, unless the SHOW option is specified.

YAML
outputs data in YAML format to the console or the file or both. This is the default.

FLAT
outputs data in a flat structure to the console or the file or both. Each line contains the whole YAML tree structure separated by slashes.

If options are specified multiple times, the last option overrides previous specifications.

Usage Notes

1. By default, the COMPEXTR utility is located on the system tools disk (MAINT 193).
2. The required CP privilege classes are ABDEG.
3. If you receive CMS Pipelines storage errors when running COMPEXTR, you may need to increase the virtual storage available in your virtual machine. To query the size of your virtual machine, issue:

```
CP QUERY VIRTUAL STORAGE
```

Increase the size of your virtual machine using the DEFINE STORAGE command, then re-IPL CMS and enter the COMPEXTR command again.

4. The CP User directory is OPTION DEVMAINT.

   Note: If an already-linked minidisk is located on the directory volume and ranges from cylinder/block 0 to at least the end of the allocated directory extent, that minidisk will be used, and no overlay minidisk will be created. In this case option DEVMAINT is not required.

5. Authority is required to connect to all TCP/IP stack servers. The userid must be added to the PERMIT or OBEY list in the PROFILE of the TCP/IP stack and VSwitch controllers. To achieve this, either add a PERMIT statement or configure the ASSORTEDPARMS option PERMITTEDUSERSONLY.
6. Note these ESM considerations for authorization to perform COMMANDS that are protected by the VMCMD CLASS. For example, if you are protecting the DEFINE.MDISK Command with VMCMD Class, you need to give the DEFINE.MDISK RACF® Profile Update access for the userid. Otherwise, the userid will not be able to use the COMPEXTR command and will receive an RPIMGR055E error.

   To give authority, use this command:

   ```
   RAC PERMIT DEFINE.MDISK CLASS(VMCMD) ID(userid) ACCESS(UPDATE)
   ```

   For more information, see *z/VM: RACF Security Server Security Administrator's Guide*.
7. Authority is required to issue the SSLADMIN QUERY STATUS DETAILS (SSL ssl server command for all SSL servers. To achieve this, add your user ID to the list of users in the :Admin_ID_list. tag ("nodeid" or SYSTEM DCPARMS file on the SSL servers).
8. To display the commands and diagnose codes you are authorized to use, enter:

   ```
   CP COMMANDS
   CP COMMANDS QUERY
   ```

   These are the CP commands used by the COMPEXTR utility:

   ```
   CP IND USER *
   CP LOCK Userid * 0 0 Logical MAP
   CP QUERY ALLOC DRCT
   CP QUERY CPLEVEL
   CP QUERY CPSERVICE APAR
   CP QUERY CPSERVICE LCLMOD ALL
   CP QUERY DASD DETAILS
   CP QUERY ENCRYPT PAGING
   CP QUERY MDISK
   CP QUERY NAMES AT *
   CP QUERY NSS
   CP QUERY RESOURCE
   CP QUERY SET
   CP QUERY SSI
   CP QUERY Userid
   CP UNLOCK Userid * 0 0 Logical
   CP VAR SYSTEM ALL
   ```

   These are the CP diagnose codes used by the COMPEXTR utility:

   ```
   DIAG04
   DIAGA0
   DIAG26C
   DIAG90
   ```
9. Note these ESM considerations for VMCMD class and diagnoses:

- If you have RACF VMCMD Class Active and you have enabled profile protection for these diagnoses:

  DIAG004
  DIAG0A0.RACONFIG
  DIAG26C

  You must issue authority to the profile, using this command:

  ```
  PERMIT DIAGnnn CLASS(VMCMD) ID(userid) ACCESS(READ)
  ```

  For more information, see *z/VM: RACF Security Server Security Administrator's Guide*.

Or

- If you do *not* have RACF profile protection enabled for DIAGNOSE X'nnn', the userid must have the correct privilege class to issue the diagnose for the specific `OPTION DIAGnnn` in the userid's CP directory entry.

10. Note these ESM considerations for authorization to perform RACROUTE functions:
    a. You must create a profile named ICHCONN in the FACILITY class. To do so, enter this command:

       ```
       RDEFINE FACILITY ICHCONN UACC (NONE)
       ```

    b. You must give UPDATE access authority to the userid you will use to issue COMPEXTR. To do so, enter this command:

       ```
       PERMIT ICHCONN CLASS (FACILITY) ID(userid) ACCESS (UPDATE)
       ```

    c. You must activate the FACILITY class if it is not already active. To do so, enter this command:

       ```
       SETROPTS CLASSACT (FACILITY)
       ```

11. For information regarding the key-value pairs polled by COMPEXTR EXEC, refer to the COMPEXTR README file, also included on the MAINT 193 minidisk. This file, in Markdown format, describes each IBM-provided key-value pair, explains potential values, and explains how to correlate this information on your z/VM system. If this file will be viewed through a web browser, change the filetype from .readme to .md.

Example

Output to the console or to a file is provided in the YAML format or as a FLAT structure (the whole YAML tree for a data point is provided in the prefix of each record). The values of the YAML output are converted to a FLAT structure as shown in the following sample output:

```
zVM:
  Identity:
    Systemname: "ZVM730"
    SSI:
      SSI_name: "n/a"
      SSI_mode: "n/a"
  Version:
    CP_level: "z/VM Version 7 Release 3.0, service level 0000 (64-bit)"
    CP_service_APAR_PTF:
    CP_service_local_modifications:
    CMS_level: "z/CMS Level 31, Service Level 0000"
  CP:
    Features:
      - Feature: "Clear_TDisk"
        Enabled: "No"
      - Feature: "Not_Disconnect_Operator"
        Enabled: "No"
      - Feature: "Password_On_Command"
        LINK:
          Enabled: "No"
        LOGON:
          Enabled: "No"
        AUTOLOG:
          Enabled: "No"
      - Feature: "Set_Privclass"
        Enabled: "Yes"
```

```
zVM/Identity/Systemname: "ZVM730"
zVM/Identity/SSI/SSI_name: "n/a"
zVM/Identity/SSI/SSI_mode: "n/a"
zVM/Version/CP_level: "z/VM Version 7 Release 3.0, service level 0000 (64-bit)"
zVM/Version/CMS_level: "z/CMS Level 31, Service Level 0000"
zVM/CP/Features/Feature$Clear_TDisk/Enabled: "No"
zVM/CP/Features/Feature$Not_Disconnect_Operator/Enabled: "No"
zVM/CP/Features/Feature$Password_On_Command/LINK/Enabled: "No"
zVM/CP/Features/Feature$Password_On_Command/LOGON/Enabled: "No"
zVM/CP/Features/Feature$Password_On_Command/AUTOLOG/Enabled: "No"
zVM/CP/Features/Feature$Set_Privclass/Enabled: "Yes"
```

Each indented item is added to the key, separated by a slash. Keys that start with a dash are added together with their value, separated by a dollar sign (included blank characters are changed to underscores).

Messages and Return Codes

Messages:

- DMS004W Warning messages issued. [RC=4]
- DMS070E Invalid parameter *parameter*. [RC=24]
- DMS653E Error executing " *option*. [RC=37]
- The CP commands or diagnose codes included in the following messages are issued by the COMPEXTR command. Some of them require CP privilege classes beyond that of a general user. If the test of the execution of a command fails, one or more of the following error messages are shown. If any of these error messages are issued, execution will be terminated.
  - DMS653E Error executing "LOCK USERID * 0 0 LOGICAL MAP". [RC=1]
  - DMS653E Error executing "UNLOCK USERID * 0 0 LOGICAL". [RC=1]
  - DMS653E Error executing "CP Query SSI". [RC=3]
  - DMS653E Error executing "CP Query ENCRYPT PAGING". [RC=3]
  - DMS653E Error executing "CP Query CPSERVICE {APAR|LCLMOD} ALL". [RC=3]
  - DMS653E Error executing "CP Query NSS". [RC=3]
  - DMS653E Error executing "RPIUCMS INIT (PERMIT to ICHCONN in class FACILITY required)". [RC=8]
  - DMS653E Error executing "NETSTAT call to DTCVSW1|DTCVSW2|DTCVSW3|DTCVSW4". [RC=<99]

- DMS653E Error executing "SSLADMIN call to SSL00001 - Cannot obtain SSL information from SSL00001 - check :Admin_ID_list. tag in DTCPARMS file". [RC=99]
- DMS653E Error executing "Diagnose 04". [RC=99]
- DMS653E Error executing "Diagnose A0 subcode 50". [RC=99]
- DMS653E Error executing "Diagnose 26C". [RC=99]
- DMS653E Error executing "Query ESM status" (Diagnose A0 subcode 8 or 70). [RC=99]
- DMSCOM653E Error executing "CP Query VAR SYSTEM ALL", rc=3
- DMSCOM653E Error executing "CP Query CPLEVEL", rc=3
- DMSCOM653E Error executing "CP Query USERID", rc=3
- DMSCOM653E Error executing "CP Query Names AT *", rc=3
- DMSCOM653E Error executing "CP IND USER *", rc=1
- DMSCOM653E Error executing "CP Query SET", rc=3
- DMSCOM653E Error executing "CP Query RESOURCE", rc=3
- DMSCOM653E Error executing "CP Query MDISK", rc=3
- DMSCOM653E Error executing "Diagnose 90", rc=99
- DMSCOM653E Error executing "Module RXCOMEXT not available", rc=99
- DMSCOM653E Error executing "Query Version (RXCOMEXT MODULE issue or wrong version)", rc=99
- DMSCOM653E Error executing "CP Define MDISK - RPIMGR055E COMMAND DEFINE.MDISK NOT DEFINED TO RACF – HCPDEF003E Invalid option - MDISK", rc=3
- DMSCOM653E Error executing "Create CP directory overlay", rc=99
- FPLFRE122E Insufficient free storage
- FPLRVR236E Too much data for variable output

Return codes:

RC
Meaning

0
Command complete — no errors

1
Not authorized

3
Not authorized

4
Warning

8
Not authorized

24
Invalid option

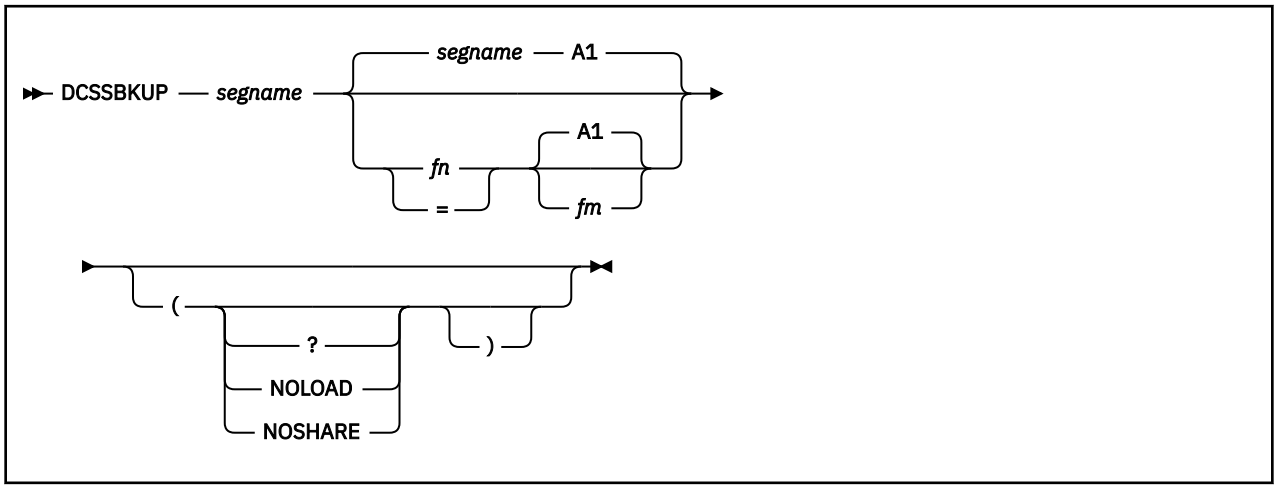
37
Invalid parameter

99
Not authorized

99
RXCOMEXT Module issue

Note: Messages and return codes may be issued because of calls to other routines or utilities. You can find a description of the message(s) you receive in *z/VM: CMS and REXX/VM Messages and Codes*.

DCSSBKUP



Authorization

Systems Programmer

Purpose

Use the DCSSBKUP utility to create a disk file containing the data from a CMS saved segment. The saved segment may later be restored using the DCSSRSAV command.

Operands

segname
is the name of the saved segment that was defined using the CP DEFSEG command. This name is used as the default file name. The file type is always DCSSBKUP. See *z/VM: CP Commands and Utilities Reference* for reference information about CP DEFSEG.

fn fm
is the optional file name and file mode of the file containing the saved segment data. The default file name is the name of the saved segment, and the default file mode is A1. The file type is always DCSSBKUP. If the file already exists, it is replaced by the new file. You may specify an equal sign (=) for the file name to default to the file name of the saved segment.

Options

?
displays the saved segment's starting and ending storage addresses and the number of pages it requires. No disk file is created.

NOLOAD
DCSSBKUP does not purge and then reload the segment before doing the backup. This allows you to backup a segment already loaded in your virtual machine.

NOSHARE
DCSSBKUP will load the segment in 'nonshared' mode before doing the backup. This allows you to backup an unmodified SW (Shared Write) mode segment that may currently be in use (and possibly modified) by another user in the system.

Usage Notes

1. By default, the DCSSBKUP utility is located on the system tools disk (MAINT 193).

2. To save a restricted saved segment, a NAMESAVE entry must be included in the user directory for each saved segment you load.
3. The file that is created by this command will contain more records than the number of pages occupied by the saved segment. The extra records are used to record the starting address, *segname*, storage protection keys, and so on. The file format is fixed (F), with a logical record length (LRECL) of 4096 characters (bytes).
4. If the backup file replaces an existing file in a shared file system (SFS) directory, it retains any of the authorities and aliases the old file had.
5. If NOLOAD and NOSHARE are both specified, NOLOAD is ignored.
6. For more information about saved segments such as creating and defining them, see *z/VM: Saved Segments Planning and Administration*.
7. To back up CP segment spaces, each individual segment member must be backed up.

Messages and Return Codes

- DMS109S Insufficient free storage available
- DMS283E The *name* saved segment could not be loaded; return code *rc* from SEGMENT
- DMS336E No saved segment name specified
- DMS394E Invalid option: *option*
- DMS671E Error creating file *fn ft fm*; rc=*nn* from *command*
- DMS2158I SEGNAME: *name*, START: *addr*, END: *addr*, #PAGES: *n*
- DMS2159E An error occured during SEGMENT FIND of segment *name*. Return code *rc* received from SEGMENT.
- DMS2159E The NOLOAD option was specified on the DCSSBKUP command but the segment *name* was not already loaded.
- DMS2210E The saved segment is not completely inside the virtual machine
- DMS2211E *segname* is a segment space and will not be backed up. To back up a segment space, the segment members must be backed up individually

Return codes:

RC
Meaning

0
Command complete — no errors

4
Segment name not specified

8
Virtual machine too large

12
Virtual storage not available

16
NOLOAD specified but segment was not already loaded

20
segname is a segment space and will not be backed up

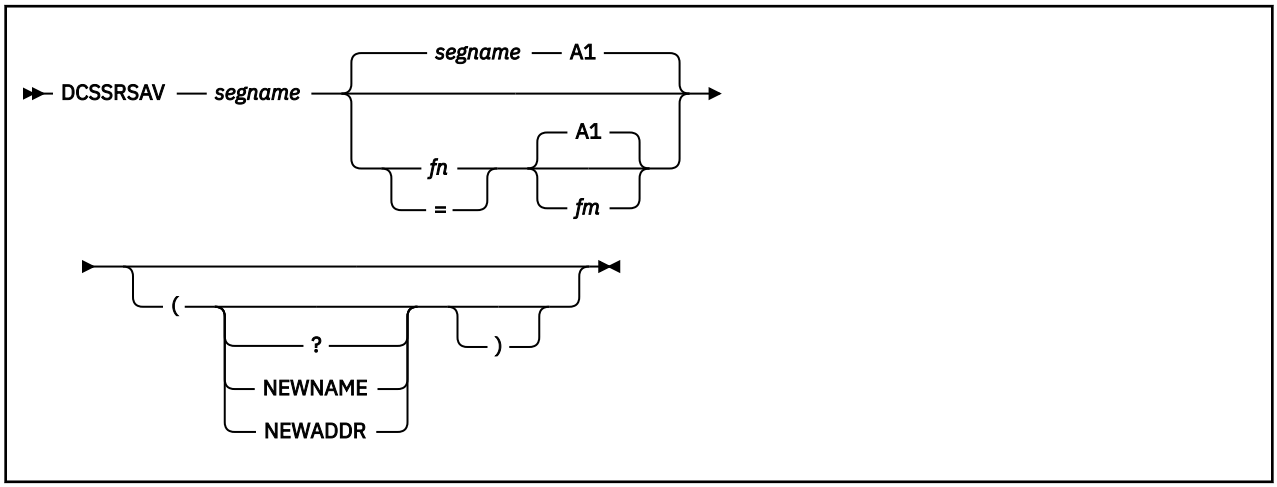
24
Invalid option

100
Explanation complete (when "?" specified)

xxx
Segment find/load error (RC from SEGMENT)

xx
Error creating DCSS backup file (RC from FSOPEN or FSWRITE)

DCSSRSAV



Authorization

Systems Programmer

Purpose

Use the DCSSRSAV utility to restore a CMS saved segment from a disk file previously backed up by the DCSSBKUP command.

Operands

segname
is the name of the saved segment to be restored. This is also the default file name of the disk file.

fn fm
is the file name and file mode of the file to be restored. The default file name is the name of the saved segment, and the default file mode is A1. The file type must always be DCSSBKUP. You may specify an equal sign (=) for the file name to default to the file name of the saved segment.

Options

?
displays the saved segment's starting address, ending address, and the number of pages its data occupies in the DCSSBKUP file. The saved segment is *not* restored.

NEWNAME
forces the use of the *segname* operand as the name of the saved segment to be restored, rather than the name contained in the DCSSBKUP file. However, the CP definition of the saved segment must still match the data within the DCSSBKUP file.

The new *segname* specified must be a predefined saved segment. See the CP DEFSEG command description in *z/VM: CP Commands and Utilities Reference* for information about defining saved segments.

NEWADDR
allows the segment to be restored to a different area from what it was backed up from. The number of hex pages in the new area being restored to must equal the number of hex pages backed up using DCSSBKUP. For example:

```
Old address   700-7FF (256 HEX PAGES)
New address range must also have 256 hex pages
```

DEFSEG must define where the new NEWADDR is, for example:

```
DEFSEG segname 540-5bf sr
```

Usage Notes

1. By default, the DCSSRSAV utility is located on the system tools disk (MAINT 193).
2. To restore a saved segment, your virtual machine must be authorized to use the CP SAVESEG command (class E by default). However, you do not need a special privilege class to use the ? option to find out information about a saved segment. See *z/VM: CP Commands and Utilities Reference* for command reference information about the SAVESEG command.
3. Your virtual storage must completely encompass the saved segment you are restoring. The following table shows some saved segment ending addresses and the corresponding amounts of virtual storage you would need to restore the saved segments.

Saved Segment Ending Address (Hex)	Virtual Storage Needed
07FFFFF	8MB
09FFFFF	10MB
0BFFFFF	12MB
0DFFFFF	14MB
0FFFFFF	16MB

4. The address area of where the segment is to be restored must be completely available for the segment. No other segments may be occupying any portion of that area.

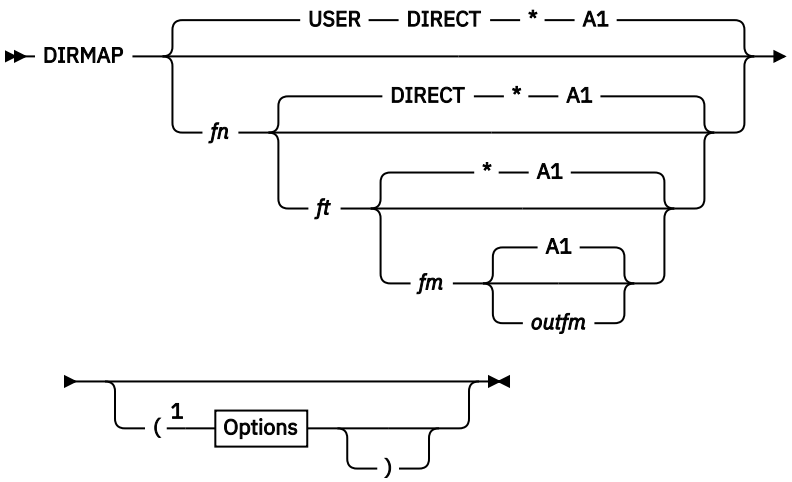
For more information about saved segments such as creating and defining them, see *z/VM: Saved Segments Planning and Administration*.

Messages and Return Codes

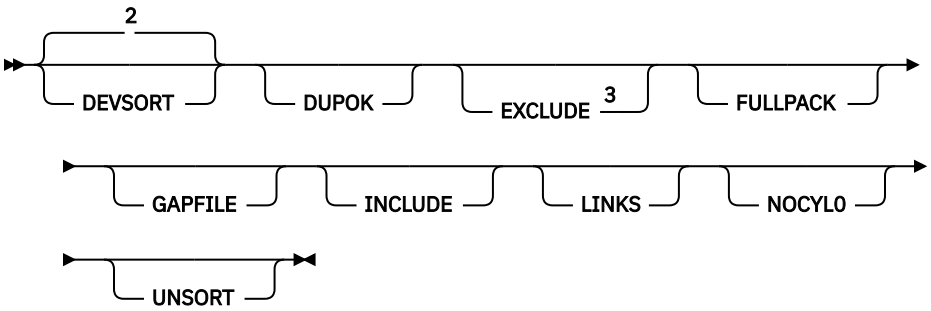
- DMS104S Error *nn* reading file *fn ft fm* from disk or directory
- DMS109S Insufficient free storage available [RC=*xx*]
- DMS283E The *name* saved segment could not be found; return code *rc* from SEGMENT [RC=*xx*]
- DMS283E The *name* saved segment could not be reserved; return code *rc* from SEGMENT [RC=*xx*]
- DMS283E The *name* saved segment could not be saved; return code *rc* from SAVESYS [RC=*xx*]
- DMS283E The *name* saved segment could not be released; return code *rc* from SEGMENT [RC=*xx*]
- DMS284E The saved segment is not completely inside the virtual machine [RC=8]
- DMS336E No saved segment name specified [RC=4]
- DMS394E Invalid option: *option* [RC=24]
- DMS2158I SEGNAME: *name*, SAVED: *mm/dd/yy*, *hh:mm:ss*
- DMS2158I START: *addr*, END: *addr*, #PAGES: *n*
- DMS2159E Invalid DCSSBKUP file format. [RC=12]
- DMS2159E The starting address in the DCSSBKUP file does not match the skeleton file's starting address. Use the NEWADDR option if this is correct. [RC=16]
- DMS2159E DCSSBKUP file segment name *name* does not match specified segment name. [RC=8]
- DMS2159E The ending address in the DCSSBKUP file does not match the skeleton file's ending address. Use the NEWADDR option if this is correct. [RC=16]
- DMS2159E Number of hex pages defined for the new address area does not match the number backed up with DCSSBKUP. See the DCSSRSAV HELPCMS file for further information. [RC=16]

- DMS2159E The starting address in the DCSSBKUP file does not match the active segment's starting address. Use the NEWADDR option if this is correct. [RC=16]
- DMS2159E The ending address in the DCSSBKUP file does not match the active segment's ending address. Use the NEWADDR option if this is correct. [RC=16]
- DMS2160I From DCSSBKUP file dated *date*.
- DMS2477T An unknown *command* failure occurred. Return code = *xx*.

DIRMAP



Options



Notes:

[1] You can enter Options in any order between the parentheses.
[2] The default is to sort by volume serial number only.
[3] EXCLUDE overrides the volume serial numbers specified by INCLUDE.

Authorization

Systems Programmer

Purpose

Use the DIRMAP utility to do MDISK/LINK mapping for the user directory. Also see the DISKMAP command in *z/VM: CP Commands and Utilities Reference*.

Operands

fn ft fm
specifies the file name, file type, and file mode of the z/VM directory DIRMAP processes. The defaults are USER DIRECT *.

outfm
is the file mode for all the output files. The default is A1.

Options

DEVSORT
specifies the MDISKMAP is to be sorted by device type first, then by volume serial number (volser). The default is to sort by volume serial number only.

DUPOK
specifies duplicate MDISKs are acceptable and are not to be treated as overlaps. A flag of 'DUP' is placed in the MDISKMAP and a message is typed on the terminal. Fullpack MDISKs are ignored for the purposes of overlap detection.

EXCLUDE
specifies volume serial numbers found in file EXCLUDE VOLSERS * are *not* to appear in the MDISKMAP. If the input file is not found, a message is issued and no volume serial numbers are excluded. If EXCLUDE is used with the INCLUDE option, EXCLUDE overrides the volume serial numbers specified by INCLUDE. See the sections on input and output files for descriptions of the file formats.

FULLPACK
specifies fullpack minidisk definitions are found in FULLPACK DEFINES. If the input file is not found, a message is issued and the default fullpack minidisk sizes are used. If the file is found, it lists new fullpack minidisk sizes for device type 3380, 3390, 9336, and FB-512 DASD. These new sizes are added to the existing fullpack minidisk sizes. Only numerics are valid for the size.

GAPFILE
specifies all gaps found are to be written in a special format to file *fn* GAPFILE. If the input file GAPFILE VOLSERS is found, it is scanned for volume serial numbers to be selected for the GAPFILE. See the sections on input and output files for descriptions of the file formats.

Volume serial numbers selected by other options (such as EXCLUDE and INCLUDE) affect this option. To select the same volume serial numbers as those selected by other options, just put the keyword ALL in the GAPFILE VOLSERS file.

INCLUDE
specifies only volume serial numbers found in the INCLUDE VOLSERS file appear in the MDISKMAP. If the input file is not found, a message is issued and all volume serial numbers are included. If INCLUDE is used with the EXCLUDE option, EXCLUDE overrides the volume serial numbers specified by INCLUDE. See the sections "DIRMAP Input Files" and "DIRMAP Output Files" below for descriptions of the file formats.

LINKS
specifies *fn* LINKMAP is to be produced. This file contains a map of all LINKs to MDISKs on the volume serial numbers processed. Any LINKs to nonexistent MDISKs produce a message on the terminal and are indicated by `*** Minidisk does not exist ***` in the LINKMAP.

NOCYL0
is for use with the GAPFILE option. It specifies any gap that starts on cylinder 0 is to be recomputed to start at cylinder 1, before being written to the GAPFILE. This is intended to prevent the allocation of MDISKs on cylinder 0, which would change the real volume serial number of the disk.

NOCYL0 can also be specified by including the NOCYL0 keyword in the GAPFILE VOLSERS file.

UNSORT
specifies the utility file DIRECT UNSORT is to be created.

DIRMAP Input Files

There are four optional input files that can be used by DIRMAP to change the content of some of the output files. Input files can reside on any accessed minidisk or any accessed shared file system (SFS) directory.

All four files follow the same format and are scanned in the same manner. The files must be fixed-block format with a logical record length of 80 and conform to the following standards:

- Comments can be included in the files. These are indicated by an asterisk in column 1.

- Fields to be scanned must be contained within columns 1–71 inclusive, and must be separated from one another by at least one blank.
- If all the required fields do not fit on a line, multiple lines can be used. However, a single field cannot span two lines.

INCLUDE VOLSERS
is used by DIRMAP to select certain MDISKs to be included in the MDISKMAP. This file is used only if the INCLUDE option is specified. Fields containing more than seven characters are ignored. However, the only seven-character *volid* allowed is &SYSRES, which represents the VM system residence volume. If the keyword ALL is found, all volume serial numbers are included in MDISKMAP.

An example of an INCLUDE VOLSERS file might be the following:

```
* This file contains a list of the packs to be included in the
* MDISKMAP report created by DIRMAP
VMRES VMTOOL
MYPACK
```

With the above input, the MDISKMAP file created only reports on volumes VMRES, VMTOOL, and MYPACK.

EXCLUDE VOLSERS
is used by DIRMAP to select certain MDISKs to be excluded from the MDISKMAP. This file is used only if the EXCLUDE option is specified. Fields containing more than seven characters are ignored. However, the only seven-character *volid* allowed is &SYSRES, which represents the VM system residence volume. If the keyword ALL is found, *no* volume serial numbers are included in MDISKMAP.

An example of an EXCLUDE VOLSERS file might be the following:

```
* This file contains a list of the packs to be excluded from the
* MDISKMAP report caused by DIRMAP
VMAUX TOOLS1
VMUSER
```

With the above input, the MDISKMAP file created does not report on volumes VMAUX TOOLS1 and VMUSER.

GAPFILE VOLSERS
is used by DIRMAP to select certain MDISKs whose GAPs are to be included in the GAPFILE. This file is used only if the GAPFILE option is specified. Fields containing more than seven characters are ignored. However, the only seven-character *volid* allowed is &SYSRES, which represents the VM system residence volume.

If the keyword ALL is found, all the GAPs contained on volume serial numbers are included in the GAPFILE. The EXCLUDE, INCLUDE, and NOCYL0 options also affect GAPFILE reporting, so ALL can be used to report GAPs for just the volume serial numbers specified by EXCLUDE, INCLUDE, or NOCYL0.

If the keyword NOCYL0 is found, gaps starting on cylinder 0 are recomputed to start at cylinder 1.

An example of an GAPFILE VOLSERS file might be the following:

```
* This file contains a list of the packs that need to have GAPS
* reported on separately by DIRMA
VMAUX TOOLS1 VMRES
```

With the above input, the DIRMAP creates a file called fn GAPFILE that contains a list of the gaps found on the above volumes.

FULLPACK DEFINES
is used by DIRMAP to list new fullpack minidisk definitions for device type 3380, 3390, 9336, and FB-512. DASD. The first word is either 3380, 3390, 9336, and FB-512. The second word is the ending minidisk cylinder number. The ending minidisk cylinder cannot be more than 8 numeric characters. All other words on the record are ignored.

Therefore, if you wanted to define a fullpack of 1000 cylinders and one of 1500 cylinders on your device type 3380, a fullpack of 2000 cylinders and one of 2500 cylinders on your device type 3390,

and a fullpack of 4280287224 blocks on both your device types FB-512 and 9336, your FULLPACK DEFINES looks similar to the following:

```
3380 999
3380 1499
3390 1999
3390 2499
9336 4280287224
FB-512 4280287224
```

Your corresponding directory statements looks similar to the following:

```
MDISK 123 3380 000000 1000 VMSRES RR
MDISK 124 3380 000000 1500 VMSPKK RR
MDISK 125 3390 000000 2000 MYPACK RR
MDISK 126 3390 000000 1500 YOURPK RR
MDISK A00 9336 14680064 4280287224 MYTEST
MDISK A01 FB-512 14680064 4280287224 YOURTS
```

DIRMAP Output Files

There are four output files that can be produced from a single run of DIRMAP. All the output files are written to the minidisk or SFS directory accessed as A, unless you specify *outfm* (see the operand *outfm*). In that case, they are written to the minidisk or SFS directory accessed at the file mode specified. If the output files are written to an SFS directory, they retain any authorities or aliases held by previous files of the same file ID.

fn MDISKMAP
is the map of all the MDISKs. This file is always produced (unless an error occurs). It should be printed with the CC option.

The format of MDISKMAP is:

```
Volser:     Volume serial number of disk
Devtype:    Device type of disk
Ownerid:    User ID of the minidisk owner
Vaddr:      Virtual address of the minidisk
Mode:       Access mode from MDISK statement
Start:      Start cylinder of minidisk or gap
End:        End cylinder of minidisk or gap
Len:        Size of minidisk in cylinders
Flags:      Identifies gaps and overlaps
Subconfig:  ID from SUBCONFIG statement or blank
Member:     System identifier from associated BUILD statement or *
```

fn LINKMAP
is the cross-reference map of all LINK statements in the user directory. This file is produced in response to the LINKS option, and should be printed with the CC option.

The format of the LINKMAP is:

```
Ownerid:    User ID of the minidisk owner
Vaddr:      Virtual address of the minidisk of Ownerid
Linkid:     Volume serial number of disk
Vaddr:      Virtual address of the linked disk for Linkid
Mode:       Access mode from LINK statement
Volser:     Volume serial number of disk
Devtype:    Device type of disk
Start:      Start cylinder of minidisk or gap
End:        End cylinder of minidisk or gap
Len:        Size of minidisk in cylinders
Subconfig:  ID from SUBCONFIG statement or blank
Member:     System identifier from associated BUILD statement or *
```

fn GAPFILE
is a list of all the gaps detected in response to the GAPFILE option.

The format of the GAPFILE is:

```
Field 1:    Volume serial number of disk
Field 2:    Device type of disk
```

```
Field 3:    Start cylinder of gap
Field 4:    End cylinder of gap
Field 5:    Number of cylinders in gap
```

This file is produced in response to the GAPFILE option.

DIRECT UNSORT
is a file produced in response to the UNSORT option.

The format of the DIRECT UNSORT file is:

```
Field 1:    Owner of minidisk
Field 2:    Virtual address of the minidisk
Field 3:    Volume serial number of disk
Field 4:    Start cylinder of minidisk
Field 5:    End cylinder of minidisk
Field 6:    Number of cylinders in minidisk
Field 7:    Device type of disk
Field 8:    ID from SUBCONFIG statement or blank
Field 9:    Member system name from associated BUILD statement or *
```

Usage Notes

1. By default, the DIRMAP utility is located on the cross release utilities disk (PMAINT 551).
2. Input files must be fixed 80-byte (F 80) format on any accessed minidisk or any accessed shared file system (SFS) directory. DIRMAP does not operate on cluster directories. It only operates on a monolithic directory file such as the USER DIRECT file or USER BACKUP file created by the Directory Maintenance Facility (DirMaint™).
3. There must be a read/write (R/W) minidisk or directory available for output files, indicated by the output file mode parameter (default A1).
4. A disk that has only fullpack minidisks is not considered to have any gaps, unless its volume serial number is in the GAPFILE VOLSERS file and the GAPFILE option is specified. When these conditions are satisfied, the complete disk is considered to be a GAP (less cylinder 0 if the NOCYL0 option was also specified). The use of the ALL keyword in the GAPFILE does not simulate the above condition, and the volume serial number is still required.
5. DIRMAP determines the size of a disk from the information it finds in the z/VM directory (device type and highest minidisk extent). If the highest minidisk extent would fit on a smaller model of the disk, DIRMAP assumes the disk is smaller than it actually is. For example, if a 3380-2 (1769 cylinders) does not have a minidisk extent higher than cylinder 884, DIRMAP assumes it is a 3380-1 (884 cylinders). To overcome this problem, just define a fullpack minidisk for the disk. If END is specified as the length of a minidisk, the ending cylinder for that disk will be determined by the other minidisks defined for that device. If END is the only length used for that device, DIRMAP will use the lowest defined cylinder length for that device type.
6. DIRMAP does not support the use of alternate tracks in the user directory. If the directory contains MDISK statements that use alternate tracks, those minidisks are flagged "OVERLAP".

   The following table shows the values DIRMAP supports for fullpack minidisks:

Table 63. Fullpack Minidisk Values

DASD Type and Model	Beginning Block or Cylinder	Total Blocks or Cylinders
3380	000	885
3380	000	1459
3380-2	000	1770
3380-3	000	2655
3390	000	455
3390	000	1084
3390-1	000	1113 See note 1 below.

Table 63. Fullpack Minidisk Values (continued)

DASD Type and Model	Beginning Block or Cylinder	Total Blocks or Cylinders
3390-2	000	2226 See note 1 below.
3390-3	000	3339 See note 1 below.
3390	000	4365
3390–9	000	65520
3390–A	000	See note 2 below.
9336	000	2147483640 See note 3 below.
FB-512	000	2147483640 See note 3 below.

Note:

a. This value also applies in 3380 emulation mode.

b. The maximum size of a CMS-formatted minidisk is 65520 cylinders. The maximum size of a minidisk on a CP-formatted extended address volume (EAV) device is 1182006 cylinders.

c. VM supports a SCSI disk, which is emulated as a 9336-020, up to a capacity of 2147483640 512-byte blocks (1 TB). Note, however, that directory, paging, and spooling allocations must reside within the first 16777216 blocks (blocks 0 to 16777215) of a CP-formatted disk.

Using the above table can help you define fullpack minidisk MDISK statements. For example, a fullpack minidisk for a 3380-2 is defined as:

```
MDISK 123 3380 000000 1770 VMSRES RR
```

7. If profile statements appear in the z/VM directory DIRMAP is processing, the profile names also appear in the fn MDISKMAP report under OWNERID and in the fn LINKMAP report under LINKID. The profile names are prefixed with a ":" to distinguish them from user IDs.
8. MDISK statements containing the DEVNO, T-DISK, or V-DISK option are not taken into account during DIRMAP processing.
9. For a given volser in the directory, if there is a mixture of device types (for example: FB-512 and 9336), the real device number is used in calculations and in the reports. For example:

   ```
   MDISK 123 FB-512 0000 100 FBARES RR
   MDISK 124 9336   0200 150 FBARES RR
   ```

   In the above example, 9336 would be used as the device type instead of FB-512.
10. If you add a fullpack minidisk extent of 999 for a 3380 DASD in your FULLPACK DEFINES file your corresponding fullpack minidisk statement in your directory can look similar to the following:

    ```
    MDISK 123 3380 000000 1000 VMSRES RR
    ```

    This support is intended for any DASD that emulates IBM DASD but has a different number of cylinders.
11. The special *volid* "&SYSRES" is allowed in a directory, but the output files will be truncated to six characters ("&SYSRE").
12. From an authorization perspective, the Ownerid value specifies the user ID to which the MDISK belongs. For multiconfiguration virtual machine instances, the MDISKs can be defined within either the identity or subconfiguration entries. In both cases the Ownerid value specifies the name from the IDENTITY statement. If the SUBCONFIG statement is not referenced by any BUILD statement, the Ownerid value contains the value *NOUSER!.
13. The SUBCONFIG value in the output is blank unless the MDISK was defined within a subconfiguration entry.
14. The MEMBER value in the output is * unless the MDISK was defined within a subconfiguration entry, in which case the value displayed is the system identifier from the BUILD statement. If there is no corresponding BUILD statement, the value is blank.

15. If a LINK statement specifies a user ID from an IDENTITY statement and a Vaddr that is defined in multiple subconfiguration entries referenced by that identity entry's BUILD statements, then the LINK is shown for every possible match.
16. If the LINK statement is defined within a subconfiguration entry, then the link is not associated with MDISKs or LINKs that are defined in subconfiguration entries for other SSI cluster members.
17. If a LINK is defined after an MDISK with the same Vaddr, the LINK is ignored. Likewise, duplicate MDISK or duplicate LINK entries are ignored.
18. DIRMAP does not perform stringent directory verification. It is assumed that all input directories are verified using DIRECTXA to be valid directories.

Examples

If the directory contains the following statements:

```
DIRECTORY SSI 123 3390 M01RES M02RES M03RES
 *
 ************************************************************
 *
 USER $DIRECT$ NOLOG
  MDISK A01 3390 001 020 M01RES R
  MDISK A02 3390 001 020 M02RES R
 *
 ****************************
 *
 *
 IDENTITY MAINT    MAINT     128M 1000M ABCDEFG
  BUILD ON TAPE1 USING SUBCONFIG MAINT-1
  BUILD ON TAPE2 USING SUBCONFIG MAINT-2

 SUBCONFIG MAINT-1
  MDISK CF1 3390 039 120 M01RES RR READ     WRITE    MULTIPLE
  MDISK CFD 3390 159 001 M01RES RR READ     WRITE    MULTIPLE
 * END OF MAINT-1

 SUBCONFIG MAINT-2
  MDISK CF1 3390 039 120 M02RES RR READ     WRITE    MULTIPLE
  MDISK CFD 3390 159 001 M02RES RR READ     WRITE    MULTIPLE
 * END OF MAINT-2
```

DIRMAP will provide the following MDISK map in the MDISKMAP output file:

```
1INSTALL  DIRECT      Map of Minidisks    14:23:24   29Sep2011 Page    1
+_______  ______  ___ __ _________   ________  _________ ____    _
0Volser Devtype  Ownerid  Vaddr Mode    Start       End        Len     Flags    Subconfig Member
+______ _______  _______  _____ ____    _____       ___        ___     _____    _________ ______

 ------------------------------------------------------------------------------------------------
 M01RES 3390                              000        000        001     Gap
                 $DIRECT$  0A01 R         001        020        020                       *
                                          021        038        018     Gap
                 MAINT     0CF1 RR        039        158        120             MAINT-1   TAPE1
                 MAINT     0CFD RR        159        159        001             MAINT-1   TAPE1
                                          160        454        295     Gap
 ------------------------------------------------------------------------------------------------
 M02RES 3390                              000        000        001     Gap
                 $DIRECT$  0A02 R         001        020        020                       *
                                          021        038        018     Gap
                 MAINT     0CF1 RR        039        158        120             MAINT-2   TAPE2
                 MAINT     0CFD RR        159        159        001             MAINT-2   TAPE2
                                          160        454        295     Gap
 ------------------------------------------------------------------------------------------------
```

Messages and Return Codes

- DMS002E File *fn ft fm* not found [RC=*xx*]
- DMS003E Invalid option: *option* [RC=112]
- DMS007E File *fn ft fm* is not fixed, 80-character records [RC=112]
- DMS104S Error *nn* reading file *fn ft fm* from disk or directory [RC=*xx*]

- DMS105S Error *nn* writing file *fn ft fm* on disk or directory [RC=*xx*]
- DMS109S Virtual storage capacity exceeded [RC=112]
- DMS2230I *Fileid* not found, *fileid* will contain all volsers.
- DMS2230I *Fileid* not found, *fileid* will not introduce any new FULLPACK definitions
- DMS2231I *Fileid* read.
- DMS2232I *Fileid* written [- no errors].
- DMS2233E No corresponding MDISK for *userid* link *addr*. [RC=104]
- DMS2234W Warning: *volume* device type *type* undefined, length unknown. [RC=104]
- DMS2235E Duplicate MDISKS on *volume*.
- DMS2236E MDISK overlap on *volume* [- end of disk overlapped]. [RC=108]
- DMS2237E No {MDISK|LINK} statement found. [RC=108]
- DMS2238E No MDISKS passed filtering - no MDISKMAP produced. [RC=108]
- DMS2239E Error detected in above statement. [RC=104]
- DMS2244W The above MDISK statement will not be processed. [RC=104]

Return codes:

RC
Meaning

0
Command complete—no errors

xx
Return code *xx* from FSOPEN, FSREAD, or FSWRITE

100
Explanation complete (when '?' specified)

104
Processing complete—minor errors

Minor errors might be one or more of the following:

- Undefined device type found
- Invalid directory statement
- LINK found without corresponding MDISK

108
Processing complete—serious errors

Serious errors might be one or more of the following:

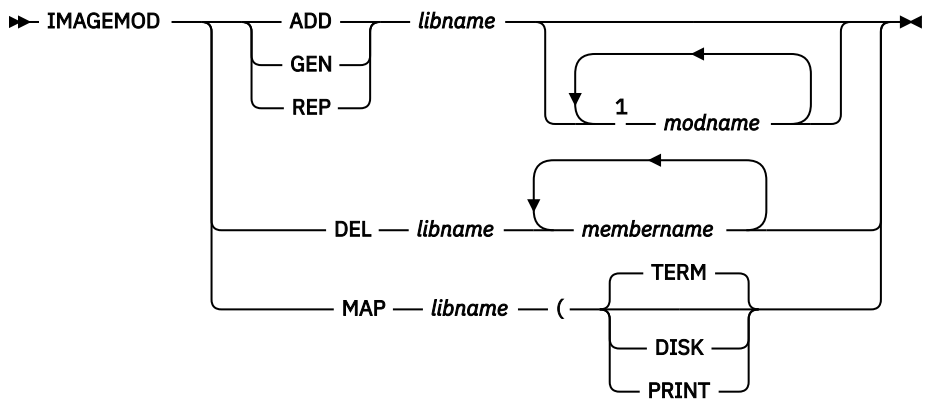
- Overlap found
- No MDISK statements found
- LINKMAP requested but no LINK statements found

112
Processing incomplete

Reasons why processing stopped might be one of the following:

- Invalid option supplied
- Insufficient free storage
- Format of input file not fixed 80

IMAGEMOD



Notes:

[1] You can specify a maximum of 10 modules.

Authorization

Systems Programmer

Purpose

Use the IMAGEMOD utility to make changes to existing image libraries. IMAGEMOD adds, deletes, or replaces individual members of an image library, or generates an entire image library.

Operands

ADD
specifies the addition of the indicated modules to an already existing image library.

GEN
causes an existing image library to be generated with the specified module names.

REP
specifies the replacement of the indicated member with a new version of that member in an already existing image library.

DEL
specifies the deletion of a member from an already existing image library. This option compresses the named system so all members are contiguous.

MAP
specifies the creation of a map of the image library on the user's terminal, virtual printer, or CMS disk. This map gives the member name, its relative byte displacement in both decimal and hexadecimal, and the byte size of the member in both decimal and hexadecimal.

Note: The MAP function does not require you specify any module names.

libname
specifies the name of the image library to be operated upon. This named system must have been created with at least one member by the IMAGELIB utility.

membername
specifies the member to be deleted from the library.

modname
specifies the name of a module (character arrangement table, copy modification, or FCB) being added or replaced. Up to ten modules may be specified on one IMAGEMOD command.

Options

TERM
specifies the map of the image library be written to the user's terminal. This is the default.

PRINT
specifies the map of the image library be written to the user's virtual printer.

DISK
specifies the map of the image library be written to a file called libname MAP A5.

For more complete information, see *z/VM: CP Commands and Utilities Reference*.

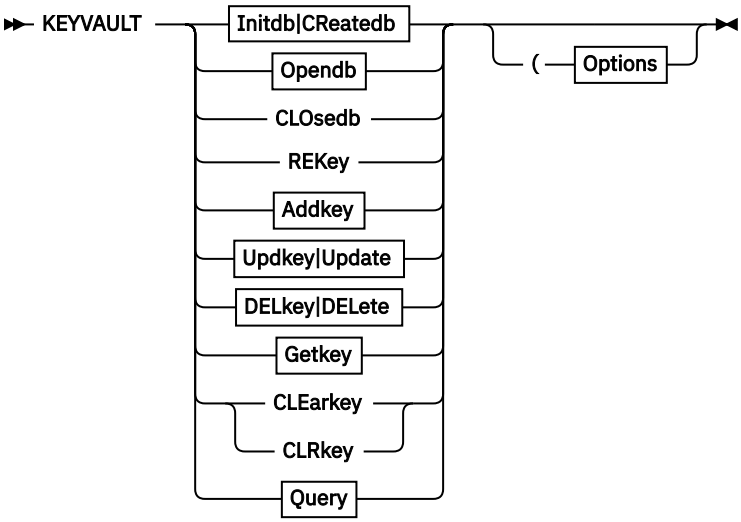
Usage Notes

1. Using VMRELOCATE for a virtual machine that might be using this function is not recommended and may have undesired results. If a virtual machine reads an image library and is then relocated to another system, any subsequent save of that image library will be written out to a different system, not the one from which it was read.

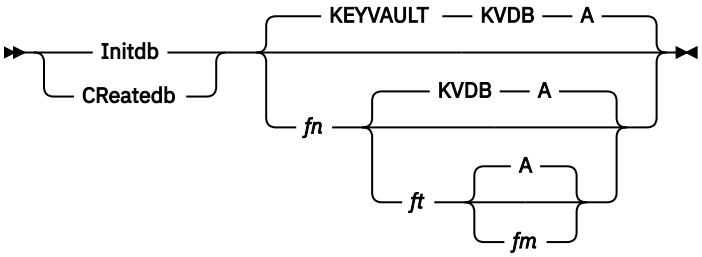
Messages and Return Codes

- DMS001E No filename specified
- DMS002E File *fn ft* not found
- DMS003E Invalid option *option*
- DMS013E Member *name* not found in library *libname*
- DMS014E Invalid function *function*
- DMS046E No library name specified
- DMS047E No function specified
- DMS109S Virtual storage capacity exceeded
- DMS346E Error *nn* loading *fn ft* from disk
- DMS347E Error *nn* loading library *libname*
- DMS348E Error *nn* saving library *libname*
- DMS349E Invalid library *libname*
- DMS751E Member *name* found in library *libname*

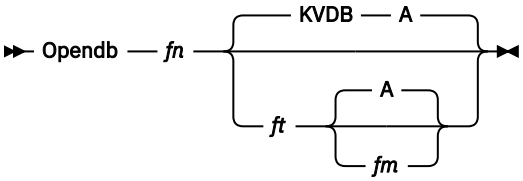
KEYVAULT



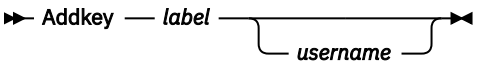
Initdb|CReatedb



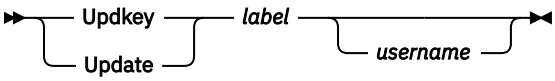
Opendb



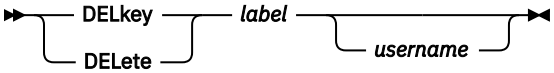
Addkey



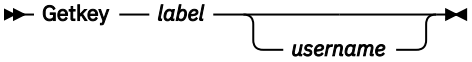
Updkey|Update



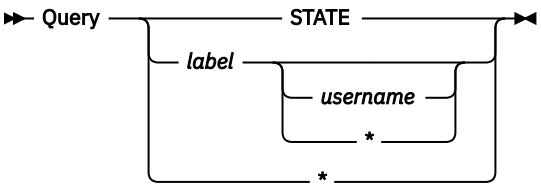
DELkey|DELete



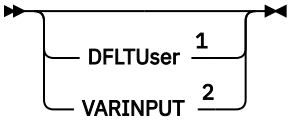
Getkey



Query



Options



Notes:

[1] The DFLTUSER option is valid with the ADDKEY, UPDATE, and UPDKEY operands only.
[2] The VARINPUT option is valid with the ADDKEY, DELKEY, INITDB, OPENDB, and UPDKEY operands (and their respective synonyms) only.

Authorization

Systems Programmer

Purpose

Use the KEYVAULT utility to encrypt key-value pairs (such as user IDs and passwords) for use in CMS programs and applications.

Operands

INITDB
CREATEDB
creates a KEYVAULT database file at the target *fn ft fm*. This database can be used to encrypt and store key-value pairs in a z/VM virtual machine. The default file is KEYVAULT KVDB A2.

During the initialization of a given database, KEYVAULT prompts the system programmer for the encryption token (a Password-Based Key Derivation Function string token) that is to be used as part of cryptographic key encryption.

OPENDB
unlocks an existing KEYVAULT database at target *fn ft fm*. To unlock the database, the system programmer is prompted for the encryption token that was used during database creation.

CLOSEDB
closes the active KEYVAULT database.

REKEY
prompts the system programmer for the current encryption token and a new encryption token, and then reenciphers the KEYVAULT database with a new key. This might be required by some installations with rules about key lifecycle and rotation.

ADDKEY
inserts a key-value pair into the KEYVAULT database in encrypted format.

label
is a unique string that helps identify key-value pairs in the KEYVAULT database.

username
is a 1- to 255-character string that represents the key portion of a key-value pair, such as a z/VM user ID.

When adding a key-value pair, you can use the DFLTUSER option to grant default operational status for key-value-related KEYVAULT operations.

UPDKEY
UPDATE
modifies the values associated with the specified key-value pair in the open KEYVAULT database.

DELKEY
DELETE
removes a key-value pair from the KEYVAULT database.

GETKEY
returns data associated with a key-value pair from the KEYVAULT database.

CLEARKEY
CLRKEY
removes the GLOBAL variables that are set by a prior GETKEY call. Before these variables are removed, their content is cleared in memory.

QUERY
confirms the presence of a key-value pair inside the KEYVAULT database, based on the parameters provided. The parameter STATE returns information about the currently-open KEYVAULT database file.

Options

DFLTUSER
For a KEYVAULT ADDKEY or UPDKEY operation, indicates that a user ID and its associated password are to be designated as the default values for a subject label database entry.

VARINPUT
Indicates that input data is to be obtained from select set of REXX variables, when the KEYVAULT command is invoked in a REXX EXEC. This option is valid for the ADDKEY, CREATEDB, DELKEY, OPENDB, and UPDKEY operands (and their respective synonyms). The REXX variables that are referenced for VARINPUT use are:

KEYVAULT_DATABASE_KEY
supplies the KEYVAULT database encryption token. An encryption token (string) with a maximum length of 130 characters can be supplied using this variable.

This variable is supported for use with the CREATEDB, INITDB, and OPENDB operations.

KEYVAULT_RECORD_USER
supplies a *username* value. A string with a maximum length of 255 characters can be supplied using this variable.

This variable is supported for use with the ADDKEY and UPDKEY operations (inclusive of synonymous operands).

KEYVAULT_RECORD_PASS
supplies a user password value. A string with a maximum length of 255 characters can be supplied using this variable.

This variable is supported for use with the ADDKEY and UPDKEY operations (inclusive of synonymous operands).

KEYVAULT_DATABASE_KEY_CURR
supplies the current KEYVAULT database encryption token. An encryption token (string) with a maximum length of 130 characters can be supplied using this variable.

This variable is supported for use with the REKEY operation only.

KEYVAULT_DATABASE_KEY_NEW
supplies a new KEYVAULT database encryption token. An encryption token (string) with a maximum length of 130 characters can be supplied using this variable.

This variable is supported for use with the REKEY operation only.

Usage Notes

1. KEYVAULT can be executed only on a z/VM system running on an IBM z14® (or later) family server. The CP Assist for Cryptographic Facilities (CPACF) feature (Feature 3863) must be enabled. zCMS must be IPLed.
2. A database is created on a per-virtual-machine basis, and is accessible only from another machine if an appropriate disk is linked.
3. After a database file has been opened for use, a change in the CMS access file mode of the minidisk or directory where it resides renders that database inoperable for reference.
4. If a database is already open when the system tries to open a KEYVAULT database file, the previously-opened KVDB file will be closed.
5. ADDKEY, DELKEY, and UPDKEY prompt for missing user ID fields if they are not provided during command execution.
6. The QUERY *label* function cannot be used without having first opened a KEYVAULT database because decryption of *username* values is necessary.
7. Not specifying a user ID with QUERY shows default users only. Not specifying a user ID with GETKEY returns the default user, if defined.
8. ADDKEY and UPDKEY set the DFLTUSER flag automatically if no default user exists for the label or if it was deleted. Adding or updating a user with the DFLTUSER option removes the existing default user flag of another user, if defined.
9. KEYVAULT handles all *label* references and values in a case-insensitive manner; all such values are translated to uppercase. Case is preserved for *username* and *password* values that are supplied or obtained from a KEYVAULT database. Any spacing used in *username* and *password* values (such as leading and trailing blanks) is also preserved.

KEYVAULT Database Storage

By default, a database file created by the KEYVAULT utility is stored on the A-disk. This file contains all stored key-value pairs the system programmer has placed inside of it.

The intent of the KEYVAULT database is to store sensitive content that should not be placed in an unencrypted CMS file: a password, passphrase, or other security-relevant token, for example. To that end, the user portion of a key-value pair under a specified label might be considered to represent a z/VM virtual machine name.

GLOBALV Variables

KEYVAULT uses these GLOBALV variables in the GLOBALV group KEYVAULT (in-storage only, not SESSION or LASTING):

KVDBFILE
contains the file ID of a KEYVAULT database file that is open for use.

PASSWORD
contains the clear-text instance of a decrypted password.

USERID
contains the clear-text instance of a decrypted user ID value, with which the subject password is associated.

KEYVAULT GLOBALV Variable/Data Example

Here is an example of in-storage KEYVAULT working variables:

```
GLOBALV SELECT KEYVAULT LIST
SELECTED TABLE IS: KEYVAULT
USERID=CSMWORK
PASSWORD=letmeinn
KVDBFILE=MYVAULT  KVDB    A2
```

Database Passphrase Validation Exit (KEYVEXIT EXEC)

The database passphrase quality might be tested against defined rules. If available, the KEYVEXIT EXEC will be called with `arg(1) = 'VALIDATE'` and `arg(2) =` *`password_phrase`* whenever a database passphrase is set or changed. If the EXEC exits with a value that starts with 0, the passphrase is accepted; otherwise, the returned value is shown and the passphrase is rejected.

Here is a sample KEYVEXIT EXEC:

```
/* KEYVEXIT EXEC */
/*====================================================================*/
/* KEYVAULT database passhprase checking exit exec.                   */
/*--------------------------------------------------------------------*/
/* This sample implements trivial checking of a KEYVAULT database     */
/* passphrase, to illustrate the input and output values which pertain */
/* to its use by the KEYVAULT utility.                                */
/*--------------------------------------------------------------------*/
/* Input:                                                             */
/*                                                                    */
/*  Arg(1)                                                            */
/*   - a call type keyword.  Currently, VALIDATE is the only call     */
/*     type made by the KEYVAULT module                               */
/*                                                                    */
/*  Arg(2)                                                            */
/*   - passphrase token or string, supplied in response to KEYVAULT   */
/*     prompt message DMSKEY2391R                                     */
/*                                                                    */
/* Output (as recognized by KEYVAULT module):                         */
/*                                                                    */
/*  return_string                                                     */
/*                                                                    */
/*   - a character return string.  When the first character returned  */
/*     is zero (0), the passphrase is accepted by KEYVAULT. Any other */
/*     first character value signifies the passphrase is to be        */
/*     rejected.  Return strings of length 99 or less are acceptable. */
/*     A string of greater length will produce a KEYVAULT validation  */
/*     exit processing error.                                         */
/*====================================================================*/
RS_lmax = 99
len_min = 5
ret_str = 0
Say '*** KEYVEXIT called ***'
Say '* Arg1: Call Keyword:' Arg(1)
Say '* Arg2: passprhase..:' Arg(2)
Say '* Arg2  length......:' Length(Arg(2))
If (Length(Arg(2)) < len_min)
  Then ret_str = '1: Length error; minimum required length is:' len_min
Exit Strip(Left(ret_str, RS_lmax))
```

Examples

1. To set up a KEYVAULT database:

   a. Issue:

   ```
   KEYVAULT CREATEDB database_fileID
   ```

   Supply and then re-enter the KEYVAULT database encryption token.

   An empty database is created and opened.

   b. Issue:

   ```
   KEYVAULT ADDKEY sample.host.name userxyz
   ```

Supply and then re-enter the associated password. The command encrypts the given user ID and password tokens, associating the with the given host name (the label) and stores the host name / encrypted user ID / encrypted password as an entry in the database file.

or:

Issue:

```
KEYVAULT ADDKEY sample.host.name
```

Enter the associated user ID. Supply and then re-enter the associated password. The command encrypts the given user ID and password tokens, associating the with the given host name (the label) and stores the host name / encrypted user ID / encrypted password as an entry in the database file.

Issue KEYVAULT ADDKEY for each system that is to be managed.

c. Issue:

```
KEYVAULT CLOSEDB
```

The database file is closed and the encryption token is cleared from memory.

2. To update a host entry password:

Issue:

```
KEYVAULT OPENDB database_fileID
```

Supply the KEYVAULT database encryption token.

With a valid encryption token, database entries can be manipulated or referenced for use.

Issue:

```
KEYVAULT UPDKEY sample.host.name useridx
```

Supply and then re-enter the new password.

The new system password for the host/username is encrypted and the old value is replaced.

Issue:

```
KEYVAULT CLOSEDB
```

The database file is closed and the encryption token is cleared from memory.

3. To update a host entry user ID and password:

Issue:

```
KEYVAULT OPENDB database_fileID
```

Supply the KEYVAULT database encryption token.

With a valid encryption token, database entries can be manipulated or referenced for use.

Issue:

```
KEYVAULT UPDKEY sample.host.name
```

Enter the new user ID. Supply and then re-enter the new password.

The new system password for the host/username is encrypted and the old value is replaced.

Issue:

```
KEYVAULT CLOSEDB
```

The database file is closed and the encryption token is cleared from memory.

4. To remove a host entry:

   Issue:

   ```
   KEYVAULT OPENDB database_fileID
   ```

   Supply the KEYVAULT database encryption token.

   With a valid encryption token, database entries can be manipulated or referenced for use.

   Issue:

   ```
   KEYVAULT DELKEY some.host.name
   ```

   The given entry is removed from the database (after validating that said entry exists); no prompting is issued to confirm deletion.

   Issue:

   ```
   KEYVAULT CLOSEDB
   ```

   The database file is closed and the encryption token is cleared from memory.

Responses

1. **KEYVAULT Query State Response Examples**

   ```
   DMSKEY2399E No key database is open.
   DMSKEY2395I Database state is open; Database file is: file_ID
   ```

2. **KEYVAULT Query Label Response Examples**

   - **Command:**

     ```
     keyvault query sample.host.com
     ```

     Responses:

     ```
     DMSKEY2405I KEYVAULT stored values:
     DMSKEY2405I Label: sample.host.com UserID: CSMWORK (default user)
     DMSKEY2408I Request completed successfully.
     ```

   - **Command:**

     ```
     keyvault query *
     ```

     Responses:

     ```
     DMSKEY2405I KEYVAULT stored values:
     DMSKEY2405I Label: sample.host.com UserID: CSMWORK (default user)
     DMSKEY2405I Label: another.system.com UserID: UserXYZ (default user)
     DMSKEY2405I Label: host.somehwere.com UserID: AdminUser (default user)
     DMSKEY2405I Label: host.somehwere.com UserID: UserABC
     DMSKEY2408I Request completed successfully.
     ```

Messages and Return Codes

- DMS2386S Dynamic area validation failed. [RC=104]
- DMS2387S Hardware feature {STFLE|MSA 3|MSA 5|zArchitecture|PCKMO Encrypt AES 256 Key|PRNO Pseudo Random Number Generator|SHA 256} not available. [RC=99]
- DMS2388S DMSCGR failed to obtain variable with return code *DMSCGR_rc* [RC=*DMSCGR_rc*]
- DMS2389E Error calling validation exit. Return code was *rc*
- DMS2390E Password phrase rejected by exit - *phrase* [RC=40]
- DMS2391R Enter [new] password phrase for database *fn ft fm*.

- DMS2391R Re-enter the same password phrase for verification.
- DMS2391R Enter an empty password phrase or one of the following keywords to cancel: CANCEL, QUIT, EXIT, or 0
- DMS2391R Enter an empty password phrase to cancel.
- DMS2391R (It will not appear when typed):
- DMS2392W Supplied password phrase values do not match.
- DMS2393S Crypto or random number failure in {klmd|prno|kmc}. [RC=99]
- DMS2394S Unknown database format. [RC=32]
- DMS2395I Database state is open; Database file is: *fn ft fm*
- DMS2396R Enter user ID:
- DMS2397E Error *LINERD_rc* from LINERD. [RC=*LINERD_rc*]
- DMS2398E A user ID is required.
- DMS2398E A password phrase is required. [RC=4]
- DMS2399E No key database is open. [RC=40]
- DMS2400E Could not find {*function*|*label*|*user*} in extended parameter list. [RC=24]
- DMS2401E User not found in database. [RC=4]
- DMS2402I Database *fn ft fm* has been {opened already|created and opened|opened|closed}. [RC=0]
- DMS2403E Incorrect password phrase supplied. [RC=76]
- DMS2404E No matching entry found in database. [RC=6]
- DMS2405I KEYVAULT stored values:
  Label: *label* UserID: *user*
  Label: *label* UserID: *user* (default user)
- DMS2406I Request canceled by user. [RC=4]
- DMS2407E User has already been defined. [RC=4]
- DMS2408I Request completed successfully. [RC=0]
- DMS2409E Database *fn ft fm* needs to be closed first. [RC=70]
- DMS2410E Password phrase for the database is too long. [RC=4]
- DMS2411E Value for variable *variable* is too long. [RC=200]
- DMS2412E Label value is too long. [RC=6]
- DMS4000E Label value of STATE is not allowed [RC=6]

Return codes:

RC
Meaning

0
Command processing completed successfully

4
Command processing stops

6
Command processing stops

32
Command processing stops

40
Command processing stops

70
Command processing stops

76
Command processing stops

99
Command processing stops

104
Command processing stops

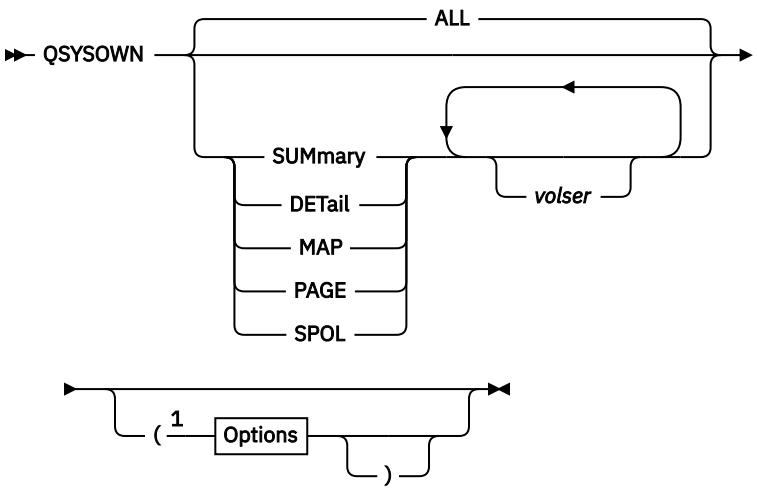
200
Command processing stops

DMSCGR_rc
Command processing stops

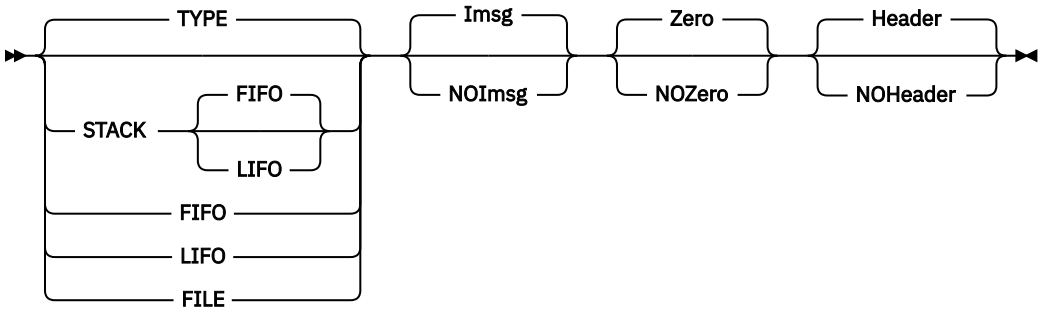
LINERD_rc
Command processing stops

Note: Messages and return codes may be issued because of calls to other routines or utilities. You can find a description of the message(s) you receive in *z/VM: CMS and REXX/VM Messages and Codes*.

QSYSOWN



Options



Notes:

[1] You can enter Options in any order between the parentheses.

Authorization

Systems Programmer

Purpose

Use the QSYSOWN utility to report the availability and use of system disk space—in particular, the CP-owned PAGE and SPOL areas on your direct access storage devices. Any DUMP space on designated DUMP volumes will be included in these reports. QSYSOWN reports on both fixed-block-architecture (FBA) and count-key-data (CKD) direct access storage devices. QSYSOWN's reports provide information you can use if you are allocating disk space, evaluating system problems, or tuning your system.

QSYSOWN supports two CP-owned *allocation types*: PAGE and SPOL. These represent your system's paging and spooling areas, respectively. The continuous disk space that makes up a PAGE and SPOL area is its *allocation extent*. CP measures the size of an allocation extent in number of pages. To create a report, you request QSYSOWN to examine the PAGE and SPOL allocation extents for a specified disk volume. QSYSOWN then returns specific details about the requested extents, including the:

- Number of pages available
- Number of pages in use
- Percent of pages in use

QSYSOWN can also summarize its detailed information by reporting the total pages and percents for your PAGE and SPOL areas and any DUMP areas system-wide. Finally, QSYSOWN can map allocation extents. If the extent is on a CKD device, QSYSOWN shows its start, end, and total cylinders. Because QSYSOWN works with both FBA and CKD storage devices, it can be used on a system that mixes these devices.

QSYSOWN can issue reports to your screen, or to an output file on your A-disk or directory. See "QSYSOWN Reports" on page 1343.

Before starting the QSYSOWN utility, make sure that you have met the necessary requirements. See usage note "2" on page 1342.

Operands

ALL
requests report data in SUMMARY form, followed by DETAIL form, for all supported allocation types. The supported allocation types are PAGE and SPOL. If a supported allocation type is not found, zeros will be listed in its related data fields. Any DUMP space will be included in the requested report. This is the default.

SUMmary
requests a data summary for all supported allocation types. A data summary lists by supported allocation type (PAGE and SPOL), the number of pages allocated and in use, and the percent of pages in use. Any DUMP space will be included in the requested report.

DETail
requests detailed data for each supported allocation type. Detailed data lists by volume serial (*volser*), address, and device allocation type (PAGE and SPOL), the number of pages allocated and in use, and the percent of pages in use. Any DUMP space will be included in the requested report.

MAP
requests a map of each supported allocation type. Map information lists by volume serial (*volser*), address, and device allocation type (PAGE and SPOL), the beginning and end of the extent, and the total number of pages (on FBA devices) or cylinders (on CKD devices) for the extent. If there are no allocations on the volume for the supported allocation type (PAGE or SPOL), an 'n/a' is listed for the start/end values and zero (0) will be returned for the total. If there are any SPOL allocations on designated DUMP volumes, these extents will be flagged as available DUMP space.

PAGE
requests data in both SUMMARY and DETAIL form for system-owned PAGE allocation extents. The allocation type PAGE includes *preferred paging areas* only. Preferred paging areas are disk areas where paged data can be written or read with the least physical motion within the device.

SPOL
requests data in both SUMMARY and DETAIL form for system-owned SPOL allocation extents. The allocation type SPOL includes non-preferred paging and spooling areas.

volser
specifies the volume serial of one or more system-owned direct access storage device volumes QSYSOWN examines for data. The default is to report on all volsers.

Options

You can specify one or more options. Enclose any option or set of options in parentheses, and place a blank between options:

STACK
specifies that, instead of going to the screen, report information be placed in the program stack in a first in, first out (FIFO) or last in, first out (LIFO) order. The default order is FIFO.

LIFO
specifies that, instead of going to the screen, report information be placed in the program stack in a last in, first out (LIFO) order. The options LIFO and STACK LIFO are the same.

FIFO
specifies that, instead of going to the screen, report information be placed in the program stack in a first in, first out (FIFO) order. The options STACK, STACK FIFO, and FIFO are the same.

FILE
specifies the report be logged in a file called QSYSOWN OUTPUT, and retained on your A-disk or directory.

TYPE
displays the data at the terminal. This is the real default for this group.

Imsg
specifies informational and warning message information be included in the report. This is the default.

NOImsg
specifies informational and warning message information be withheld from the report.

Zero
specifies that if a supported system-owned allocation type is not found, that type will appear in the report with zeros in its related data fields. This is the default.

NOZero
specifies that if a supported system-owned allocation type is not found, that type will be omitted from the report. For example, if no PAGE or SPOL allocations are defined on your system, these types will not appear in your summary, detail, or map reports.

Header
specifies report titles and column identifiers will be included in the report. This is the default.

NOHeader
specifies the report will contain data only; report titles and column identifiers will be omitted. Informational and warning messages follow QSYSOWN's regular report, unless you specify NOIMSG.

Usage Notes

1. By default, the QSYSOWN utility is located on the system tools disk (MAINT 193).
2. The virtual machine running the QSYSOWN utility requires:
   - CP user privilege class C or E, and class D, to examine real storage.
   - Optionally, user privilege class B to obtain real disk access storage device addresses (using the QUERY DASD command). If the virtual machine using QSYSOWN does not have privilege class B, QSYSOWN substitutes n/a in the volume address field of the report.
   - 5 MB of virtual storage.
3. The unit of measure used in the map report depends on the disk device type. For fixed-blocked-architecture (FBA) devices, the unit of measure is expressed in pages; for count-key-data (CKD) devices the unit of measure is in cylinders.
4. If the first cylinder on a CKD device is allocated as either PAGE or SPOL space, the detailed information obtained for this extent will not include the space reserved by CP on this first cylinder. The number of pages available and in-use data shown in both the Detail and Summary reports will be adjusted accordingly.
5. Any DUMP space available or in-use on system volumes designated as DUMP will be included in the requested summary, detail, and map reports. The usage of SPOL and DUMP space is correctly calculated and will appear separately in the summary and detail reports. If there is no DUMP space found in-use, it will not appear in these reports. The map report flags SPOL allocations as *DUMP* to show DUMP space available on system volumes.
6. In a QSYSOWN summary or detail report (see “QSYSOWN Reports” on page 1343), the value of the 'Allocd' field (total pages allocated) and the 'In-use" field (total pages in use) can be as follows:
   - If the number of pages is greater than 9,999,999, *pppppp* is shown, where *pppppp*K is the number of pages rounded to the nearest thousand.

- If the number of pages is greater than 999,999,999 and less than 999,999,999,999, *ppppppM* is shown, which is the number of pages rounded to the nearest million.
- If the number of pages is greater than 999,999,999,999, *ppppppG* is shown, which is the number of pages rounded to the nearest billion.

QSYSOWN Reports

When you issue a QSYSOWN request, QSYSOWN returns your requested data, along with related messages, in a report. QSYSOWN sends the report to your screen. If you use the FILE option, QSYSOWN places the information in a file named QSYSOWN OUTPUT, and retains it on your minidisk or directory accessed as file mode A. If you use the STACK, FIFO, or LIFO options, QSYSOWN returns the requested information to the program stack instead of your screen.

The following examples show sample QSYSOWN reports and the commands that created them. These reports were obtained from a mixed system using both FBA and CKD direct access storage devices. Note that in a map report (like the one shown in Figure 77 on page 1344), QSYSOWN maps FBA devices in pages and CKD devices in cylinders.

If you issue the command:

```
qsysown summary (noimsg
```

QSYSOWN returns a summary report to your screen in this format:

```
** Summary Information:

          Total-Pages
Type    Allocd   In-Use   %-Used
--------------------------------
PAGE     54600     3385      6.2
SPOL     54500     5657     10.4
```

Figure 75. Sample QSYSOWN Summary Report—Mixed System

If you issue the command:

```
qsysown detail (noimsg
```

QSYSOWN returns a detail report to your screen in this format:

```
** Detail Information:

                                Total-Pages
Volser  Addr  Device  Type   Allocd   In-Use   %-Used
-----------------------------------------------------
SC3390  024D  3390    PAGE    12000        0      0.0
                      SPOL    12000     1164      9.7
SC9336  0104  9336    PAGE    15000        0      0.0
                      SPOL    11000        0      0.0
VM3390  01B2  3390    SPOL     9600        0      0.0
                      PAGE     9600        0      0.0
ZVMRES  01A6  3380    SPOL    21900     4493     20.5
                      PAGE    18000     3385     18.8
```

Figure 76. Sample QSYSOWN Detail Report—Mixed System

If you issue the command:

```
qsysown map (noimsg
```

QSYSOWN returns a map report in this format:

```
** Map Information:

                                 Cylinder
Volser  Addr  Device  Type   Start     End   Total
-------------------------------------------------
SC3390  024D  3390    PAGE     250     349     100
                      SPOL     350     449     100
SC9336  0104  9336    SPOL   30000   44999   15000
                      PAGE   45000   55999   11000
VM3390  01B2  3390    SPOL     350     449     100
                      PAGE     450     549     100
SP5RES  01A6  3380    SPOL     200     345     146
                      PAGE     349     468     120
ZVM001  02FF  3380    SPOL     n/a     n/a       0
                      PAGE     n/a     n/a       0
```

Figure 77. Sample QSYSOWN Map Report—Mixed System

Note: The start and end values of 'n/a' indicate the PAGE and SPOL space are not allocated for the specified Volser.

QSYSOWN places messages at the end of a report, following the summary, detail, or map information. Messages you receive by means of the IMSG option will be presented like this:

```
** Informational/Warning Messages:

Volser, VMDSK1, is not a valid system owned volume
Volser, VMDSK2, is not a valid system owned volume
System owned volume not mounted: VMSAVE
System owned volume not mounted: SC3390
```

Figure 78. Sample QSYSOWN Message Information—Mixed System

Invoking QSYSOWN Remotely

If you are the central site administrator for a distributed system, and you want to get system-owned DASD information about a remote system, you can do so by using the Programmable Operator Facility to invoke QSYSOWN.

First, make sure you have the Programmable Operator Facility running on the remote system to be queried, and the user ID running the Programmable Operator Facility has access to the MAINT 193 minidisk, where QSYSOWN resides. Also ensure you are authorized in the Programmable Operator Facility RTABLE to issue the CMD command. Then, by issuing a command from your console at the central site, you can request the Programmable Operator Facility to invoke QSYSOWN on the remote system, and route the information back to your screen.

The command is:

```
TELL userid AT nodeid CMD QSYSOWN operands (options
```

where:

userid
is the user ID of the Programmable Operator Facility machine running on the remote system.

nodeid
is the node ID of the remote system.

operands
are the QSYSOWN operands you want to use

options
are the QSYSOWN options you want to use.

Note: Do not use the STACK, LIFO, FIFO, and FILE options if you want the QSYSOWN information routed to the central site and displayed on the screen where you entered the command.

To save your QSYSOWN information, you can use the FILE option to file the data instead of sending it to your screen. However, the data file will not be routed to your user ID at the central site automatically. You will need to write a Programmable Operator Facility action routine to do this for you.

For more information on the Programmable Operator Facility, see *z/VM: CMS Planning and Administration*.

Messages and Return Codes

- DMS109S Insufficient free storage available [RC=98]
- DMS278E Unable to set requested language *langid* [RC=1*xxx*]
- DMS514E Return code *nn* from *command* [RC=2xxx|3xxx|4xxx]
- DMS2413W QSYSOWN requires additional privilege class D for this release of VM. [RC=11]
- DMS2421W System owned volume not mounted: *volid*.
- DMS2423W This user is not authorized to obtain real volume addresses.
- DMS2425T No *type* allocations found
- DMS2425T No *type* allocations found on the specified volumes.
- DMS2426T The QSYSOWN module must be NUCXLOADed prior to invocation. [RC=90]
- DMS2427T The QSYSOWN module must be called from the REXX environment. [RC=4|91]
- DMS2428T A required stem name is missing on QSYSOWN invocation. [RC=92]
- DMS2429T QSYSOWN variable stem names cannot exceed *num* characters. [RC=93]
- DMS2431T The CP system symbol table is missing required information. [RC=95]
- DMS2434T Symbol table read failed RC=*rc*. [RC=115]
- DMS2435T This user is not running in the required VM CP environment. [RC=10]
- DMS2436T Invalid *parm* specified: *parm*. [RC=24]
- DMS2437T Cannot select both options *parm* and *parm*. [RC=26]
- DMS2438W No allocation map for system owned volume: *volid*.
- DMS2444T This user lacks the privilege to display real storage. [RC=11]
- DMS2447T The *command* MODULE is not accessible to this user. [RC=28]
- DMS2460E *Command* failed. Return code = *rc.*. [RC=8]

Return codes:

RC
Meaning

4
No EXECCOM entry point found

8
Error from EXECCOM

10
Not running in required VM CP environment

11
Not authorized to examine real storage

24
Invalid calling argument

26
Option error

28
Missing the required routine QSYSOWN MODULE

90
No matching system-owned allocations found

91
EPLIST error

92
Parameter list error

93
Parameter list error

95
System symbol table error

98
Insufficient free storage

100
Explanation complete (when '?' specified)

115
System symbol table error

1xxx
Return code *xxx* from SET LANGUAGE

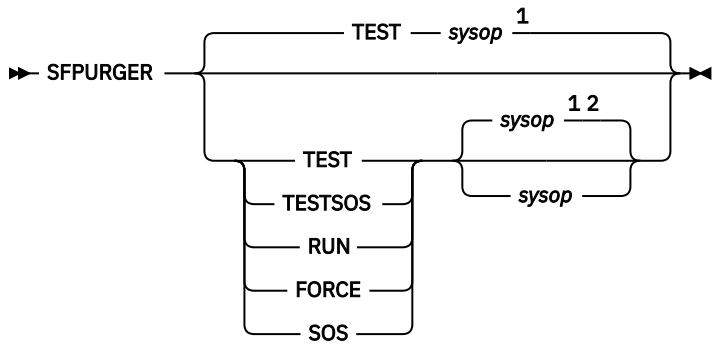
2xxx
Return code *xxx* from NUCXLOAD

3xxx
Return code *xxx* from XMITMSG

4xxx
Return code *xxx* from EXECIO

Note: Messages and return codes may be issued because of calls to other routines or utilities. You can find a description of the message(s) you receive in *z/VM: CMS and REXX/VM Messages and Codes*.

SFPURGER



Notes:

[1] For TEST and TESTSOS, *sysop* defaults to the user ID of the virtual machine running SFPURGER.
[2] For RUN, FORCE, and SOS, *sysop* defaults to OPERATOR.

Authorization

Systems Programmer

Purpose

Use the SFPURGER utility to manage your spool space and spool files. You can purge, ignore, or put one of several kinds of holds on a spool file. You can also handle a spool file as instructed by a user-written action routine. Together, these SFPURGER activities can be described as *spool file maintenance*.

SFPURGER performs spool file maintenance using instructions you give it ahead of time, at intervals you determine. You provide the instructions to SFPURGER using options and control files. You can use SFPURGER locally, or use it in the distributed environment to manage spool files on unattended systems. SFPURGER is most useful as an automated tool that runs unattended from a dedicated virtual machine.

For information on the virtual machine requirements for running SFPURGER, the SFPRUGER options and control files, and the output files that SFPURGER creates, see "SFPURGER: Requirements, Setup, and Use" on page 1352.

Operands

TEST
tests SFPURGER, *without actually performing spool file maintenance*. SFPURGER processes your SFPURGER OPTIONS file (if you create one) and your SFPURGER CONTROL file, just as in RUN mode. SFPURGER also creates output files as in RUN mode. Using the information in the output files, you can track SFPURGER's processing, and verify your options and control files work as you intended. This is the default.

TESTSOS
tests SFPURGER the same way TEST mode does, but instead processes the *emergency* control file, SOS CONTROL. As with TEST mode, you can use SFPURGER's output file information to verify your options and control files work correctly, *without actually performing spool file maintenance*.

RUN
executes SFPURGER for normal operation. SFPURGER reads the SFPURGER CONTROL file and does spool file maintenance, *unless*—

1. It is invoked during prime shift, *or*
2. It is invoked after running successfully in RUN, FORCE, or SOS mode earlier the same day.

If either 1 or 2 occurs, SFPURGER displays a message at the console and ends processing.

FORCE
executes SFPURGER, which reads the SFPURGER CONTROL file and does spool file maintenance, *regardless* of the time it is invoked or whether it has run successfully earlier the same day.

SOS
executes the same way FORCE mode does, but the control file SFPURGER reads is SOS CONTROL. You can use SOS mode to do *emergency* spool file maintenance.

sysop
is a 1- to 8-character user ID you assign to receive important SFPURGER progress and error messages. *sysop* defaults to OPERATOR, unless the mode of operation is TEST or TESTSOS, in which case *sysop* is forced to the user ID of the virtual machine running SFPURGER. You can specify *sysop* only if you specify TEST, TESTSOS, RUN, FORCE, or SOS.

The RUN, LOG, and TST files will be assumed to be from 19*xx* if *xx* is a year after 70. If *xx* is 70 or before, they will be assumed to be from 20*xx*.

Usage Notes

1. By default, the SFPURGER utility is located on the system tools disk (MAINT 193).
2. CP user privilege class D, E, and G are required. Optionally, class B is required to issue the MSGNOH command.

   Note: Class E users might want to specify the IGNORE action for named saved system (NSS) files in the SFPURGER CONTROL file to avoid purging them.
3. If the system is protected by an ESM, the virtual machine running SFPURGER requires UPDATE access to all user's virtual readers.
4. When specifying the IGNORE action in the SFPURGER CONTROL file, sufficient keywords should be used to fully describe a group of spool files to ensure they will not be affected by a later control file statement.
5. When specifying the QUEUE keyword in the SFPURGER CONTROL file, the valid queue choices are PRT (printer), PUN (punch), RDR (reader), IMG (image library), NLS (message repository), NSS (named saved system or saved segment), TRF (system trace), and UCR (user class restructure).
6. When specifying the TYPE keyword in the SFPURGER CONTROL file, the valid spool files choices are CON (console), DMP (dump), PRT, PUN, RDR, IMG, NLS, NSS, TRF, and UCR.
7. If SFPURGER matches hold-related actions to system data (IMG, NLS, NSS, TRF, and UCR) spool files due to the results from processing the SFPURGER CONTROL file, these actions will be changed to IGNORE; hold-related actions are not supported for system data spool files because of CP limitations.
8. SFPURGER checks both the opening and closing dates for files and uses the older of the two dates when determining what files to purge. This can cause confusion when dealing with CP dump files. CP dump files are created on the system's dump processor user ID when the system was started or when the last CP dump was taken and will sit in the dump processor's reader until a CP dump (RESTART dump, SNAPDUMP, etc.) is taken. Because a CP dump might not be taken over a long period of time, the opening date on this file can be quite old. If the SFPURGER CONTROL file indicates to purge dump files after a shorter period of time, this dump file will be purged the first time that SFPURGER runs. A way to handle this situation is to use the ORIGINID, USERID, etc. keywords in the SFPURGER CONTROL file. The sample SFPURGER CONTROL file shipped with the product contains sample code of one way to handle this situation:

   ```
   * Ignore dump files that have an origin of SYSTEM. Purge any other
   * dump files after 4 weeks.  Ignor the rest
   TYPE DMP     ORIGINID SYSTEM                    ACTION IGNORE
   TYPE DMP     DAYS 29                            ACTION PURGE
   TYPE DMP                                        ACTION IGNORE
   ```

   With this coding, SFPURGER will ignore any dump files that have an origin of SYSTEM (which CP dump files do), purge any other dump files older than 29 days, and ignore any other dump files.

Note: This same situation holds true for any files, not just CP dump files. However, most files are opened and closed in the same day so this situation does not occur.

9. Support for the user class restructure (UCR) function and the OVERRIDE utility have been removed. If any UCR files exist on the system, they can be queried and manipulated, but the contents of those files will not be processed by CP.

Messages and Return Codes

- DMS2442E This user lacks the privilege class for MSGTYPE MSGNOH.
- DMS2443I MSGTYPE has been set to MSG.
- DMS2444T This user lacks the privilege to display real storage. [RC=36]
- DMS2445T This user lacks the privilege to handle SYSTEM spool files. [RC=36]
- DMS2446E The user userid is an invalid destination for console log files. The log file will not be sent.
- DMS2447T The *command* MODULE is not accessible to this user. [RC=44]
- DMS2448T SFPURGER was invoked with invalid parameter *parm*. [RC=40]
- DMS2449I No files purged.
- DMS2450E The SYSOP value *value* is invalid.
- DMS2451I SYSOP has been set to OPERATOR.
- DMS2452I SFPURGER starting at *hh:mm:ss* on *dd/mm/yy*.
- DMS2453I Running in *type* mode - *ft*.
- DMS2454I You cannot invoke SFPURGER RUN in prime shift, *hh:mm:ss* - *hh:mm:ss*. [RC=0]
- DMS2455I You cannot invoke SFPURGER RUN twice in one day. [RC=0]
- DMS2456I Erasing old output files till *yydddd*.
- DMS2457I Output files are not erased in *type* mode.
- DMS2458T Error number *rc*. [RC=4|8|12|16|28|32|>1000]
- DMS2459I Examining output file ...
- DMS2460E *command* failed. Return code = *rc*.
- DMS2461I *type* mode - scanning only.
- DMS2462I Spool file scanning begins ...
- DMS2463I *num* of the *total* spool files {HAVE|WOULD have} been purged.
- DMS2464W Return code *rc* was received from *command*.
- DMS2465I SFPURGER *type* has ended.
- DMS2466I Run terminating - Return code *rc*.
- DMS2467I No action taken.
- DMS2468I SFPURGER run ends.
- DMS2469I SFPURGER OPTIONS file processed ...
- DMS2470I Using *command* MODULE with *fn* CONTROL file.
- DMS2471I Increase virtual storage and try again.
- DMS2472I Rectify error in control file.
- DMS2473I Decrease spooling activity.
- DMS2474I Contact Systems Support for advice.
- DMS2475I Check parameters and try again.
- DMS2476I Rectify disk errors and retry.
- DMS2477T An unknown *command* failure occurred. Return code = *rc*.
- DMS2478T The *parm* value *value* in the SFPURGER OPTIONS file is invalid. [RC=20]

- DMS2479T The option *parm* in the SFPURGER OPTIONS file is invalid. [RC=20]
- DMS2480I Rectify error in OPTIONS file and try again.
- DMS2481E The action *parm* in the control file is unknown.
- DMS2482I {Executing|Testing}: *command_string*
- DMS2483I Appending *fn* CONTROL file to *fn* CONTROL file.
- DMS2484I The node control file to append, *fn* CONTROL, does not exist.
- DMS2485I *num* of the *total* spool files {HAVE|WOULD have} been changed.
- DMS2486I *num* of the *total* spool files {HAVE|WOULD have} been handled by user exits.
- DMS2487I Reason code *num record*.
- DMS2488E A control file record must end in an action.
- DMS2489S SFPURGER is terminating due to previous errors. [RC=8]
- DMS2490E The *fn* CONTROL file cannot be found. [RC=28]
- DMS2491E There is insufficient free storage to run SFPURGER. [RC=4]
- DMS2492E A control file record cannot start with an action.
- DMS2493S There is an error in the ACTSECT card logical chaining. [RC=16]
- DMS2494S An invalid keyword *parm* was specified in a control file record.
- DMS2495E Invalid data *parm* was specified in a control file record.
- DMS2496I Control card scan complete.
- DMS2497S SFPURGER cannot run due to changing spool file chains. [RC=12]
- DMS2498S SFPURGER had a program check. Code is *code* PSWADDR *addr*. [RC=32]
- DMS2499E SFPURGER abend - dumping.

Return codes:

RC
Meaning

0
Normal exit

This includes exits resulting from an unsuccessful invocation either during prime shift or after an earlier run the same day.

4
Unable to allocate storage

8
Error found in control file

12
Spool file chains keep changing

16
Internal logic error

20
Error found in SFPURGER OPTIONS file

28
Control file not found

32
Program check

This error should be accompanied by a VMDUMP, and the dump file should be sent to your system support center for evaluation. Before doing so, check that the abend is not caused by incorrect privilege classes.

36
Missing privilege class

40
Invalid parameter on SFPURGER invocation

44
Missing SFPURGER MODULE

48
Unable to find sort module

100
Explanation complete (when '?' specified)

xxx
Error xxx from SORT MODULE (+100)

1xxx
Return code xxx from FSREAD

2xxx
Return code xxx from FSWRITE

SFPURGER: Requirements, Setup, and Use

The SFPURGER utility manages spool space and spool files. SFPURGER is usually set up to run unattended on a dedicated virtual machine. SFPURGER performs spool file maintenance using instructions you give it ahead of time, at intervals you determine. You provide the instructions to SFPURGER using options and control files, and SFPURGER keeps a record of its processing in a set of output files.

Virtual Machine Requirements

The virtual machine running SFPURGER requires:

- CP user privilege class D, E, and G. Class B is required if you want to issue the MSGNOH command.

  Note: Class E users might want to specify the IGNORE action for named saved system (NSS) files in the SFPURGER CONTROL file to avoid purging them.
- UPDATE access to all users' virtual readers if the system is protected by an ESM.
- EMSG set to ON or TEXT. Your message format depends on the EMSG setting used on the virtual machine running SFPURGER. If EMSG is set to OFF, you will see only the most severe error messages. If EMSG is set to TEXT, you will see the text of the other messages, but not their message identifiers. Message identifiers help you locate supplemental message information. See *z/VM: CMS and REXX/VM Messages and Codes* for more information. If EMSG is set to ON, you will see all messages and identifiers.

  When checking your SFPURGER input files for errors (using TEST or TESTSOS mode), set EMSG ON to receive all messages and identifiers. For other operations you may prefer to set EMSG to TEXT. This simplifies your console log by eliminating unwanted message identifiers, while the most severe error messages will still have them. (See *z/VM: CP Commands and Utilities Reference* for more information on the SET EMSG command.)
- 5 MB of virtual storage.
- Reader, printer, and punch devices.
- An A-disk or directory with space enough to store the SFPURGER work and output files. Disk space can vary, however. The length of time you keep your SFPURGER output files will be one determining factor. The size of these output files is another factor. File size itself is determined by the number of spool files on your system; output files will be larger when you have a large volume of spool files being processed.

SFPURGER Input Files

Before you run SFPURGER, you need to create input files tailored to your own installation:

SFPURGER OPTIONS
The file that specifies how SFPURGER will work at your installation.

SFPURGER CONTROL
The file that specifies the actions you want performed on your spool files.

SFPURGER OPTIONS File

You do not need to create an options file if you are satisfied with the defaults provided by SFPURGER (see Table 64 on page 1353). If you use the defaults, no options file will exist.

If you want to tailor SFPURGER to your installation, you can create an options file called SFPURGER OPTIONS. SFPURGER looks for an SFPURGER OPTIONS file when it runs. Although only one options file is used, the options in it can be changed between runs, should your system requirements change. If SFPURGER does not find this file during processing, it uses its default options. Unlike the control files used by SFPURGER, the SFPURGER OPTIONS file cannot be renamed.

Creating the SFPURGER OPTIONS File

Using the SFPURGER options listed in Table 64 on page 1353, you can tailor SFPURGER to your system by:

- Defining your prime shift
- Selecting your message format
- Naming your own control and module files
- Specifying the user who receives your LOG files

Statements in an SFPURGER OPTIONS file consist of an option and a value or values. For example, an option statement telling SFPURGER to send a LOG file to user ADMIN at node CENTRAL is:

```
CONSOLE  ADMIN CENTRAL
```

The following table describes the options and their values and defaults.

Table 64. SFPURGER OPTIONS Options

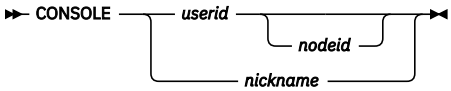
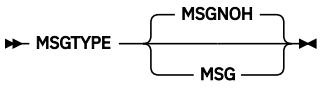
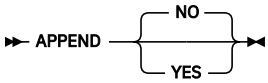
OPTION	Description
	Defines the user ID to which SFPURGER sends its LOG file. Optionally, a node ID may be specified after the user ID, to send the LOG file to a user ID at a specific node. As an alternative to naming a user ID, you can specify a *nickname* defined in the NAMES file on the virtual machine running SFPURGER: `CONSOLE` *nickname* If you use the CONSOLE option, make it the first option statement in your file. Then, if subsequent options fail, the user whose user ID is specified in the CONSOLE option will still get a LOG file to evaluate. **If you are working with centrally-managed systems,** use the CONSOLE option to ensure your central system receives all pertinent console logs from its distributed systems. Then, whenever SFPURGER runs, the LOG file will be forwarded to a designated user ID at the central system. **The default is *not* to send a LOG file**.
`KEEPDAY` *days*	Specifies the number of days, from 1 to 365, you want to keep your old LOG, RUN, and TST files before SFPURGER erases them. Each time it executes in RUN, FORCE, or SOS mode, SFPURGER uses the current KEEPDAY value to examine the output files from previous runs, and decide which ones to erase. **The default is 14**.
	Indicates whether messages will be issued to the *sysop* user ID (see "SFPURGER" on page 1347 for the definition of this user ID) with a four-line time stamp and prefix header. Use MSG to get a time stamp and header, or MSGNOH to get a briefer one-line message, with no time stamp and header. SFPURGER will ignore a MSGNOH value if the user ID running SFPURGER does not have CP privilege class B. **The default is MSGNOH—**however, this value will be forced to MSG if the user ID running SFPURGER does not have privilege class B.
`PRIMSHFT` *hh:mm:ss hh:mm:ss*	Defines the beginning and end of your prime shift. Your local system time and the 24-hour-clock are used. SFPURGER will not operate in RUN mode during prime shift hours. **The default is 08:00:00 18:00:00**.
`SORTMOD` *filename*	Identifies the file name of the CMS sort module (file type, MODULE) used by SFPURGER. This option allows you to rename the sort module, using a file name of your choice. **The default is SORT**.
`SFPCNTL` *filename*	Identifies the file name of the control file (file type, CONTROL) SFPURGER uses during RUN, FORCE, and TEST mode. It also identifies the file name of the LOG, RUN, and TST files produced by this control file. This option allows you to rename the regular control file, using a file name of your choice. **The default is SFPURGER**.

Table 64. SFPURGER OPTIONS Options (continued)	
OPTION	**Description**
SOSCNTL *filename*	Identifies the file name of the control file (file type, CONTROL) SFPURGER uses during SOS or TESTSOS mode. It also identifies the file name of the LOG, RUN, and TST files produced by this control file. This option allows you to rename the emergency control file, using a file name of your choice. **The default is SOS**.
SFPMOD *filename*	Identifies the file name of the module (file type, MODULE) called by SFPURGER to process spool files. This option allows you to rename the module, using a file name of your choice. **The default is SFPURGER**.
	Specifies if a control file named *nodeid* CONTROL should be appended to the regular CONTROL file. **If you are working with centrally-managed systems**, you might use this option. If a node has spool files that require some extra handling, create a node control file that describes that extra handling. Format the node control file just like SFPURGER CONTROL. Give it a file name the same name as the node ID, a file type of CONTROL, and place it on the node whose spool files require the extra handling. When it runs, SFPURGER will append the node control file to SFPURGER CONTROL. SFPURGER will then perform spool file maintenance as directed by the two files. (Each node can have only *one* node control file.) **The default is NO**.

OPTIONS File Format

In your options file, you can have no more than 132 characters on a line; blank lines and lines that begin with an asterisk (*) are ignored. Each option statement can have only one option, and you must specify both an option and a value in the statement. If you use the CONSOLE option, remember to make it the first option statement in your file. Then, if subsequent options fail, the user whose user ID is specified in the CONSOLE option will still get a LOG file to evaluate.

The following sample options file shows how tailoring may be done. Use it as a guide in preparing an options file for your installation. The file name and file type must be SFPURGER OPTIONS.

```
*************************************************************************
*                                                                       *
*                     Sample SFPURGER OPTIONS File                      *
*                                                                       *
*                                                                       *
*************************************************************************

* Send console log to user ID ADMIN at node CENTRAL
CONSOLE   ADMIN CENTRAL

* Erase LOG and RUN files that are more than 3 days old
KEEPDAY  3

* Set prime shift start and end times
PRIMSHFT  07:30:00 16:30:00

* Use defaults for the following:
*   MSGTYPE  SORTMOD  SFPCNTL  SOSCNTL  SFPMOD  APPEND
```

Figure 79. Sample SFPURGER OPTIONS File

SFPURGER CONTROL File

Before you can run SFPURGER, you must create at least one control file. SFPURGER uses this file for normal operations in RUN mode, and also in FORCE and TEST modes. You may additionally create an emergency control file for use in SOS (and TESTSOS) mode.

When invoked in TEST, RUN, or FORCE mode, SFPURGER uses the control file specified by the SFPCNTL option in the options file. The SFPCNTL option lets you give your regular control file a file name of your choice. If you omit the SFPCNTL option, SFPURGER looks for the control file with the default file name SFPURGER. The file type must always be CONTROL.

Similarly, the SOSCNTL option in the options file lets you rename the emergency control file SFPURGER uses in TEST or TESTSOS mode. If you omit the SOSCNTL option, SFPURGER will look for the emergency control file with the default file name SOS (file type, CONTROL).

Creating the SFPURGER CONTROL File

The SFPURGER control file contains a series of statements, each consisting of a keyword or keyword group and one related action. The keywords describe a set of spool files, and the action specifies the action for SFPURGER to take on each spool file in the set. The action can ignore, purge, or put one of several kinds of holds on a spool file. It can also handle a spool file as instructed by an action routine you create, SFPX*nnnn* (see Table 67 on page 1357).

Control statements can have one or more keywords for each action. For example, a one-keyword control statement telling SFPURGER to purge any class 0 spool files is:

```
CLASS 0  ACTION PURGE
```

Some control statements may require additional keywords to tell SFPURGER which spool files to act on. Here is a three-keyword control statement:

```
QUEUE RDR  ORIGINID MAINT  USERID DEMO1     ACTION SFPX01
```

The statement tells SFPURGER to invoke action routine SFPX01 to handle reader files that user MAINT sent to user DEMO1.

During SFPURGER processing, every spool file is checked against the keyword/action statements in the control file. The first statement whose keyword a spool file matches will decide the action for that file. Whenever a spool-file-to-keyword comparison does not match, the rest of the control statement is skipped and SFPURGER goes to the next statement, and so on, until a match is found. Once it finds a match, SFPURGER skips the rest of the control file and moves on to match the next spool file. If no match is found, SFPURGER assigns an IGNORE action to the spool file.

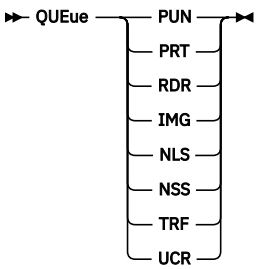
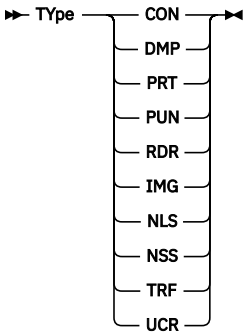
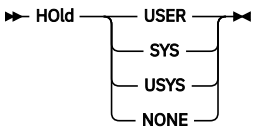
The following table describes the keywords and their values. The minimum abbreviation you can enter for each keyword is shown in uppercase.

Table 65. SFPURGER CONTROL Keywords

KEYWORD	Description

`USERid` *userid*	Specifies the current spool file owner.
	Specify an equal sign (=) for *userid*, to indicate spool files owned by the user ID of the virtual machine running SFPURGER:
	`USERid =`
	To request a specific subset of user IDs, you can use two special characters in the *userid* value. Use an asterisk (*) to represent any 0–7 characters in the user ID. For example, you can specify `abc*` for *userid*, to indicate all spool files owned by user IDs beginning with `abc`:
	`USERid abc*`
	Use a percent sign (%) to represent any single character in the user ID. For example, you can specify `%abc%` for *userid*, to indicate all spool files owned by five-character user IDs whose three middle characters are `abc`:
	`USERid %abc%`
	You can also use asterisks and percent signs in combination. For example, you can specify `%abc*` for *userid*, to indicate all spool files owned by user IDs whose second, third, and fourth characters are `abc`:
	`USERid %abc*`
	For more information on using these two special characters—* and %—see “Using Pattern Matching to Specify Sets of Files” on page 14.

Table 65. SFPURGER CONTROL Keywords (continued)

KEYWORD	Description
ORIGinid *userid*	Specifies the spool file originator. To request a specific subset of user IDs, you can use the two special characters * and % when specifying the *userid* value. See the USERID description in this table for details on using these special characters.
	Specifies the spool file's queue. The valid queue choices are the punch, printer, reader, image library, message repository, named saved system, system trace, and user class restructure queues.
	Specifies the type of spool file. The valid spool file choices are the console, dump, printer, punch, reader, image library, message repository, named saved system, system trace, and user class restructure spool files.
DAYs *days*	Specifies a number from 1 to 365, representing the minimum number of days the spool file can exist before it is processed by SFPURGER.
CLass *class*	Specifies with a single character the spool file class. The valid spool file choices are the letters A-Z and the numbers 0-9. No special character can be specified.
FName *filename*	Specifies the spool file's file name. To request a specific subset of file names, you can use the two special characters * and % when specifying *filename*. See the USERID description in this table for details on using these special characters.
FType *filetype*	Specifies the spool file's file type. To request a specific subset of file types, you can use the two special characters * and % when specifying *filetype*. See the USERID description in this table for details on using these special characters.
	Specifies a type of hold. The valid hold choices are the user, system, user and system, and no holds.
RECords *recs*	Specifies a 1 - 12 digit number, representing the minimum number of records a spool file must have before it is processed by SFPURGER.

The following table indicates the actions SFPURGER can take. The minimum abbreviation you can enter for each action is shown in uppercase.

Table 66. SFPURGER CONTROL Actions	
ACTION	**Description**
`ACTion IGNore`	Ignores the spool file. This action prevents a spool file from being processed again by another control statement below the one containing the IGNORE. Once an IGNORE has occurred, no further action will be taken on the spool file.
`ACTion PURge`	Purges the spool file.
`ACTion USERHold`	Places the spool file in USERHOLD.
`ACTion SYSHold`	Places the spool file in SYSHOLD.
`ACTion USYSHold`	Places the spool file in USERHOLD and SYSHOLD.
`ACTion UNOHold`	Takes the spool file out of USERHOLD.
`ACTion SNOHold`	Takes the spool file out of SYSHOLD.
`ACTion NOHold`	Takes the spool file out of USERHOLD and SYSHOLD.

PI

Table 67. SFPURGER CONTROL Actions	
ACTION	**Description**
`ACTion SFPX`*nnnn*	Invokes an SFPX*nnnn* routine you have written to handle one or more spool files in a way not provided for by SFPURGER's available actions. For example, you might write an SFPX*nnnn* routine to transfer a spool file to another user. The SFPX*nnnn* routine can be either an EXEC or a MODULE; any ambiguities regarding file type are resolved in the CMS environment, using the standard CMS search order. The module or exec is invoked by SFPURGER with the parameters: `*SFPX owner queue spoolid` where: ***owner*** is the spool file owner's user ID ***queue*** is the spool file's queue name (RDR, PRT, PUN, IMG, NLS, NSS, TRF, UCR) ***spoolid*** is the spool file's spool file number The routine should return a zero return code upon successful completion. A nonzero return code causes an error message to be sent to the console and the operator. The error message indicates the routine name and the return code. The first four characters of the user action routine must be SFPX; the *nnnn* portion can be any valid CMS file name 1- to 4-character suffix. **Use of an SFPX*nnnn* routine is optional**.

PI end

CONTROL File Format

The control file can have no more than 132 characters on a line. Control statements can include as many keywords and keyword values as line length allows, but only one action. The action delimits the end of the line; any data that follows the action will be considered a comment. To save processing time, begin with control statements most likely to produce spool file matches. Blank lines are ignored, and lines beginning with an asterisk (*) are treated as comments.

A sample control file follows. Use the following sample control file as a guide in preparing a control file for your installation.

```
*************************************************************************
*                                                                       *
*                    Sample SFPURGER CONTROL File                       *
*                                                                       *
*                                                                       *
*************************************************************************
                                                              Statement No.
* Ignore any spool files found in the NSS queue (privilege class E)
QUEUE NSS                                  ACTION IGNORE             1

* Purge any spool files found in class 0
CLASS 0                                    ACTION PURGE              2

* Purge listing output with 1000 or more records
RECORDS 1000   FTYPE LISTING               ACTION PURGE              3

* Keep spool files owned by maintenance user IDs
USERID MAINT*                              ACTION IGNORE             4

* Ignore dump files that have an origin of SYSTEM. Purge any other
* dump files after 4 weeks.  Ignore the rest
TYPE DMP       ORIGINID SYSTEM             ACTION IGNORE             5
TYPE DMP       DAYS 29                     ACTION PURGE              6
TYPE DMP                                   ACTION IGNORE             7

* Purge files awaiting transmission after 2 weeks.  Ignore the rest
QUEUE RDR      USERID RSCS     DAYS 15     ACTION PURGE              8
QUEUE RDR      USERID RSCS                 ACTION IGNORE             9

* Purge files received through RSCS after 4 weeks.  Ignore the rest
QUEUE RDR      ORIGINID RSCS   DAYS 29     ACTION PURGE             10
QUEUE RDR      ORIGINID RSCS               ACTION IGNORE            11

* Change console logs to system hold after 1 week
TYPE CON       DAYS 8                     ACTION SYSHOLD            12

* Invoke SFPX01 to handle reader files that MAINT sent to DEMO1
QUEUE RDR      ORIGINID MAINT  USERID DEMO1   ACTION SFPX01         13

* Purge any reader files in USERHOLD after 4 weeks.  Ignore the rest
QUEUE RDR      DAYS 28         HOLD USER   ACTION PURGE             14
QUEUE RDR                                  ACTION IGNORE            15

* Purge any other print files after 2 weeks.  Change the rest
* to USERHOLD
QUEUE PRT      DAYS 15                     ACTION PURGE             16
QUEUE PRT                                  ACTION USERHOLD          17

* Purge any other punch files after 1 week.  Ignore the rest
QUEUE PUN      DAYS 8                      ACTION PURGE             18
QUEUE PUN                                  ACTION IGNORE            19
```

Figure 80. Sample SFPURGER CONTROL File. The statement numbers that appear in bold type in this figure are for your reference only; they are not part of the file itself.

Output from SFPURGER

As SFPURGER runs, it writes progress messages to the console. The most important of these are sent to the *sysop* user ID (see "SFPURGER" on page 1347 for the definition of this user ID). To keep a record of the processing, SFPURGER creates three output files. Each output file takes its file name from the CONTROL file that produced it. The file type is either LOG*yynnn*, RUN*yynnn*, or TST*yynnn*.

SFPURGER LOG*yynnn*
: The console log file for the run that occurred on Julian date *yynnn*. The LOG file contains a record of activities that occurred during the run.

SFPURGER RUN*yynnn*
: The file that relates SFPURGER actions to the individual spool files processed during the run that occurred on Julian date *yynnn*. The RUN file identifies each spool file by file name, file type, and spool file number (spoolid). A RUN file can result only from invoking SFPURGER in RUN, FORCE, or SOS mode.

SFPURGER TST*yynnn*
The file that results from invoking SFPURGER in TEST or TESTSOS mode contains the same kind of information as the RUN file. In other words, a TST file is simply a RUN file produced in a test mode, and then renamed.

Additionally, a no-run console log, **SFPURGER NOR***yynnn*, will appear instead of SFPURGER LOG*yynnn* if SFPURGER fails in RUN mode. To get this log, the failure must result from SFPURGER being invoked during prime shift, or twice in one day.

When you run SFPURGER, you will always get a LOG or NOR output file. Getting a RUN or TST file will depend on the success of your SFPURGER execution. If it completes successfully, you will get a RUN or TST file with your LOG file.

SFPURGER LOG*yynnn* File

Content and Purpose. The LOG file provides a record of console messages and spool file processing activities, and some tabulations summarizing the spool file maintenance for the run. Using the LOG file you can track SFPURGER's progress as it executed. If SFPURGER failed, you can determine the point of failure from the progress and error messages listed in the LOG file.

Tracking SFPURGER Processing. When SFPURGER runs, it first processes your OPTIONS file, if you created one. At the same time, SFPURGER begins writing to the LOG file a list of the progress messages sent to *sysop* like those shown in Figure 81 on page 1360.

Next, SFPURGER processes the CONTROL file. As it does, SFPURGER assigns a reason code to each valid keyword/action statement. A reason code is just a way to reference a spool file action with a number. SFPURGER assigns reason codes to your CONTROL file statements in order from the top down: The first keyword/action statement receives reason code 1, the second statement receives reason code 2, and so on. SFPURGER then lists in the RUN (or TST) file each keyword/action statement together with its assigned reason code like those shown in Figure 81 on page 1360.

To ensure no spool file gets missed during a run, SFPURGER adds a "catch-all" entry to its control file processing. This entry appears as the *last reason code* in the reason code list. The catch-all entry assigns an IGNORE action to any spool file not handled by one of the control file statements.

Citing numbers from the reason code list, SFPURGER justifies the action it intends to take on every spool file in the current run. This information appears in a spool file actions table, which follows the reason code list in the RUN (or TST) file.

Next, actual spool file maintenance is performed, as directed by the RUN file. (With a TST file, spool file maintenance that *would* have occurred is noted, but *not* performed.) As each spool file action is executed (or noted), SFPURGER writes to the LOG file the action it took (or would have taken). IGNORE actions are *not* logged, however. Only records of spool files that actually changed (or that would have changed) during processing are written to the LOG file.

Finally, SFPURGER performs some tabulations. At the end of the LOG file SFPURGER records the total number of spool files processed in the current run, and of these, the number of files that have (or would have) been purged, changed, or presented to exit routines (your SFPX*nnnn* routines). These tabulations provide a short summary of the spool file maintenance that occurred (or that would have occurred) during the run.

Destination. The LOG file is saved on the A-disk or directory of the virtual machine running SFPURGER, until erased by some later SFPURGER run. (See the KEEPDAY option in Table 64 on page 1353 for details about output file erasure.)

The LOG file is saved in packed format, and may have to be unpacked before it can be viewed (for information about the packed format, see "COPYFILE" on page 79). The LOG file is also sent to the user ID specified by the CONSOLE option in the options file.

Here is a sample SFPURGER LOG*yynnn* file produced in RUN mode:

```
**                              **                                **
**                        SFPURGER Console Log                    **
**         Created by z/VM 5741-A09     10 Feb 2001   18:30:09
**                           ADMIN at CENTRAL                     **
**                              **                                **

SFPURGER OPTIONS file processed …                        ◄ Progress Messages
SFPURGER starting at 18:29:59 on 10 Feb 2001
Running in RUN Mode - RUN01041
Using SFPURGER MODULE with SFPURGER CONTROL file
Erasing old output files till 01038

Control card scan complete.

Examining output file …                                  ◄ Spool File Processing
Spool file scanning begins ...
Executing: CP PURGE MAINT PRT 0022
0001 FILE  PURGED
Executing: CP PURGE MAINT PUN 0030
0001 FILE  PURGED
Executing: SFPX01 *SFPX DEMO1 RDR 0014
0001 FILE  TRANSFERRED
Executing: CP CHANGE DEMO2 PRT 0019 HOLD
0001 FILE  CHANGED
Executing: CP CHANGE DEMO2 PRT 0021 HOLD
0001 FILE  CHANGED
2 of the 8 spool files HAVE been purged                      ◄ Summary/Tabulations
2 of the 8 spool files HAVE been changed
1 of the 8 spool files HAVE been handled by user exits
```

Figure 81. Sample SFPURGER LOG yynnn File. The pointers that appear in bold type in this figure are for your reference only; they are not part of the file itself.

A LOG file produced in TEST or TESTSOS mode will differ slightly from the LOG file in Figure 81 on page 1360, which was produced in RUN mode. Some differences: Because no spool file maintenance is actually performed during TEST and TESTSOS modes, in their LOG files a progress message will say "`Output files are not erased in TEST mode`", and their processing statements will say SFPURGER is only "`Testing`" rather than "`Executing`" spool file actions. In addition, their summary tabulations will show the spool file actions that "`WOULD have been`" done (in RUN, FORCE, or SOS mode), rather than the actions that "`HAVE been`" done, as in Figure 81 on page 1360.

SFPURGER RUN*yynnn* File

Content and Purpose. The RUN file is produced when SFPURGER runs successfully in RUN, FORCE, or SOS mode. When SFPURGER assigns all the keyword/action statements in your SFPURGER CONTROL file a reason code, it lists them in the RUN file. Following this reason code list, SFPURGER provides a table showing every unopened spool file that was on the system when SFPURGER ran, and the intended action for each, as specified in your CONTROL file. It also shows the reason code governing the intended action. The RUN file can be used to track, by reason code, the processing of a specific spool ID back to *one* keyword/action statement in the CONTROL file.

Note: As a precaution, SFPURGER does not process spool files being peeked or read at the time it runs; consequently, these "open" spool files are not in the RUN file list.

Tracking Spool File Processing. When creating a RUN file, SFPURGER lists each unopened spool file present at the time of the run, the action to be taken on the spool file, and reason code for that action. Accordingly, you can find out what happened to a spool file by:

1. Obtaining its reason code *number* from the table in the RUN file. This number references the processing for a specific spool ID.
2. Finding the matching reason code *text*: All the reason codes and their related text appear in a reason code list at the top of the RUN file. The reason code text indicates the keyword and action to which the reason code number was matched. SFPURGER got this text from a keyword/action statement in the CONTROL file.

3. Confirming that the reason code has been matched to the correct keyword/action statement in the CONTROL file. Do this by checking the reason code number against the sequence of control statements in the CONTROL file. Count the control statements from the top down (do not count blank lines or comment lines). Note the number of the control statement whose keyword and action match the reason code text. That number and the reason code number should be the same.

For example, look at the first spool ID, 0020, in the spool file actions table in the sample SFPURGER RUN*yynnn* file (Figure 82 on page 1361). Note that the reason code defining file 0020's processing is reason code 14. Now find reason code 14 in the reason code list just above the table. The reason code 15 text, `QUEUE RDR ACTION IGNORE`, should match the fifteenth control statement (from the top) in the sample SFPURGER CONTROL file (Figure 80 on page 1358), which it does.

By following these steps and tracking the spool file's handling from the RUN file back to the CONTROL file, you get a complete *audit trail*—a record of processing—for the spool file. The RUN file completes this audit trail by showing what happened to *all* unopen spool files on your system that were processed by SFPURGER—including the *ignored* spool files, which are not detailed in the LOG file.

Destination. The SFPURGER RUN*yynnn* file is saved on the A-disk or directory of the virtual machine running SFPURGER, until erased by some later SFPURGER run. (See the KEEPDAY option in Table 64 on page 1353 for details about output file erasure.)

The RUN file is saved in packed format, and might have to be unpacked before it can be viewed (for information about the packed format, see "COPYFILE" on page 79). The RUN file is *not* sent to the user ID specified by the CONSOLE option in the options file.

Here is a sample SFPURGER RUN*yynnn* file produced in RUN mode:

```
**                              **                                      **
**                       SFPURGER Run File                              **
**         Created by z/VM 5741-A09     10 Feb 2001   18:30:09
**                           ADMIN at CENTRAL                           **
**                              **                                      **
Reason code 1 QUEUE NSS ACTION IGNORE.                                  ◄ Reason Code List
Reason code 2 CLASS 0 ACTION PURGE.
Reason code 3 RECORDS 1000 FTYPE LISTING ACTION PURGE.
Reason code 4 USERID MAINT* ACTION IGNORE.
Reason code 5 TYPE DMP ORIGINID SYSTEM ACTION IGNORE.
Reason code 6 TYPE DMP DAYS 29 ACTION PURGE.
Reason code 7 TYPE DMP ACTION IGNORE.
Reason code 8 QUEUE RDR USERID RSCS DAYS 15 ACTION PURGE.
Reason code 9 QUEUE RDR USERID RSCS ACTION IGNORE.
Reason code 10 QUEUE RDR ORIGINID RSCS DAYS 29 ACTION PURGE.
Reason code 11 QUEUE RDR ORIGINID RSCS ACTION IGNORE.
Reason code 12 TYPE CON DAYS 8 ACTION SYSHOLD.
Reason code 13 QUEUE RDR ORIGINID MAINT USERID DEMO1 ACTION SFPX01.
Reason code 14 QUEUE RDR DAYS 28 HOLD USER ACTION PURGE.
Reason code 15 QUEUE RDR ACTION IGNORE.
Reason code 16 QUEUE PRT DAYS 15 ACTION PURGE.
Reason code 17 QUEUE PRT ACTION USERHOLD.
Reason code 18 QUEUE PUN DAYS 8 ACTION PURGE.
Reason code 19 QUEUE PUN ACTION IGNORE.
Reason code 20 Queue *** Action Ignore.                                 ◄ Catch-All Entry

                                ▼ Spool File Actions Table ▼

          Action    Owner    Queue    Spool ID   Reason Code No.     File name   File type

SFP100I   IGNORE    DEMO1     RDR       0020      Reason    15        MEETING     MEMO
SFP100I   IGNORE    DEMO2     RDR       0018      Reason    15        CMS         EXEC
SFP100I   IGNORE    MAINT     PRT       0016      Reason    04        TEST1       EXEC
SFP100I   PURGE     MAINT     PRT       0022      Reason    03        TEST1       LISTING
SFP100I   PURGE     MAINT     PUN       0030      Reason    02        TEST        EXEC
SFP100I   SFPX01    DEMO1     RDR       0014      Reason    13        REPORT      OUTPUT
SFP100I   USERHOLD  DEMO2     PRT       0019      Reason    17        REP01       SCRIPT
SFP100I   USERHOLD  DEMO2     PRT       0021      Reason    17        TR123       LISTING
```

Figure 82. Sample SFPURGER RUN yynnn File. The pointers that appear in bold type in this figure are for your reference only; they are not part of the file itself.

SFPURGER TST*yynnn* File

Content and Purpose. The TST file is produced when SFPURGER runs successfully in a test mode (TEST, TESTSOS). Otherwise it works exactly like the RUN file, which is produced in a non-test mode. Like the RUN file, the TST file lists all the keyword/action statements in the current control file, and assigns each of these statements a reason code. It further provides a table showing the intended action for each spool file, and the reason code governing that action.

In the test modes that produce the TST file, no spool file maintenance is done. The purpose of the test modes and TST file is to help verify your options and control files work correctly, before you use a mode of operation where spool file maintenance actually *is* done. After a test run, you can use the TST file to verify your spool file handling by following the same procedure as outlined in "SFPURGER RUNyynnn File" on page 1360.

Destination. The SFPURGER TST*yynnn* file is saved on the A-disk or directory of the virtual machine running SFPURGER, until erased by some later SFPURGER run. (See the KEEPDAY option in Table 64 on page 1353 for details about output file erasure.)

The TST file is saved in packed format, and may have to be unpacked before it can be viewed (for information about the packed format, see "COPYFILE" on page 79). The TST file is *not* sent to the user ID specified by the CONSOLE option in the options file.

A sample TST file would look exactly like the sample RUN file in Figure 82 on page 1361.

SFPURGER NOR*yynnn* File

Content and Purpose. The NOR (no-run) file is a short console log ending with an error message. When SFPURGER is invoked in RUN mode, you might get a NOR file instead of a LOG and RUN file. This happens only when SFPURGER fails because it was invoked during prime shift, or twice in one day. The NOR file notifies you of the failure.

Format and Destination. The file name of the NOR file is the same as that of the control file that produced it, and the file type is NOR*yynnn*, where *yynnn* is the Julian date. The NOR file is sent to the user ID specified in the CONSOLE option (if one is specified), and then erased. It is *not* retained on the A-disk or directory of the virtual machine running SFPURGER.

When to Run SFPURGER

First Do a Test Run. Use TEST mode the first time you invoke SFPURGER on your system. This allows you to see if your SFPURGER OPTIONS file (if you have one) and your SFPURGER CONTROL file work correctly. Progress and informational messages in the output files will help you to find any errors.

Afterward, whenever you change your input files, do a TEST run to verify SFPURGER's correct operation before attempting normal operation in RUN mode, or unscheduled operation in FORCE mode. Similarly, a run in TESTSOS mode should be done whenever you create or change an SOS CONTROL file, before using SOS mode.

In TEST mode, no spool files are changed. You may want to follow up a successful TEST run with a normal run, to perform actual spool file maintenance.

Normal Operations. For normal operation, invoke SFPURGER at regular intervals in RUN mode; for instance, at the start or end of each day. This enables SFPURGER to run when spool files are relatively stable. In particular, you want to run SFPURGER when programs like RSCS and DIRMAINT, which manipulate a large number of spool files, are not logged on or are idle. SFPURGER will not run when the spool files keep changing.

SFPURGER is most useful as an automated tool that runs unattended from a dedicated disconnected virtual machine. You may set up SFPURGER to be invoked from a disconnected virtual machine by setting up a WAKEUP TIMES file and using the WAKEUP utility. For information on how to set up a disconnected virtual machine to invoke programs at timed intervals, see "WAKEUP" on page 1202.

Unscheduled Operations. To run SFPURGER at an unscheduled time (such as prime shift) but perform *standard* spool file maintenance as specified by SFPURGER CONTROL, use FORCE mode.

Note: If you run SFPURGER more than once the same day, the output files from the latest run will replace those of the previous run. SFPURGER replaces them because you cannot keep on your A-disk or directory multiple files with the same file identifier. When SFPURGER replaces one output file with another, any authorities associated with the replaced file will *not be transferred* to the new file.

Emergency Operations. Occasionally, "emergency" situations might arise where you want to run SFPURGER at an unscheduled time and perform *nonstandard* spool file maintenance. In such situations you can make SFPURGER run immediately, using SOS mode and an alternate control file, SOS CONTROL, which you have prepared ahead of time to address the special circumstances.

For example, if your system's spool space is nearly depleted or if you are running out of spool IDs, you may want to purge some additional spool files—spool files not keyed for purging in your SFPURGER CONTROL file. To do so, invoke SFPURGER in SOS mode. This allows SFPURGER to run regardless of the time of day, or whether or not it has run successfully earlier the same day. SFPURGER then uses your SOS CONTROL file to take whatever specific actions you have indicated.

Reminder: Test a new or changed SOS CONTROL file by running SFPURGER in TESTSOS mode, before attempting emergency operation in SOS mode.

Beginning Spool File Maintenance

To use SFPURGER, proceed as follows:

1. Determine the privilege classes you will require for the virtual machine running SFPURGER (privilege class B is optional), and implement them on your system.
2. Create your SFPURGER OPTIONS file (if you use one) and SFPURGER CONTROL files. Create the SFPX*nnnn* action routines you want. If you need a node control file, create one.
3. Test your OPTIONS and CONTROL files by invoking SFPURGER in TEST mode, and also test your SOS CONTROL file (if you made one) by invoking SFPURGER in TESTSOS mode.
4. Set up SFPURGER to run at regular intervals from an exec, or invoke it from the command line in the CMS environment.

Note: If you are working with centrally-managed systems, follow the steps above for *each system* where SFPURGER will run.

SYSWATCH

Authorization

Systems Programmer

Purpose

Use the SYSWATCH utility to collect system performance and other data from centrally-managed systems, and to forward it to the central site. The collected data can be displayed on a series of panels to help support personnel detect problems before users at the monitored systems encounter them. SYSWATCH can also be used in the single system environment.

Before running SYSWATCH, you need to set up the required virtual machines, execs, and input files. See “SYSWATCH: Requirements, Setup, and Use” on page 1370.

If the system environment has been prepared, you can run the SYSWATCH EXEC on any authorized user's virtual machine to display the SYSWATCH panels.

Usage Notes

1. By default, the SYSWATCH utility is located on the system tools disk (MAINT 193).
2. If the sample exit routines are used, CP privilege classes D and G are required. Privilege class E is required when monitoring CP INDICATE LOAD data.

Using the SYSWATCH Panels

SYSWATCH uses several levels of panels to display its data. Node IDs of centrally-managed systems are presented on the primary, or System Selection, panel. From the System Selection panel you can choose the System Detail panel (by selecting a node ID and pressing Enter) to see system data, or the Exception Colors panel (by pressing PF6) to change the highlight colors for the various **warning levels**. Warning levels are limits you set for specific types of data, such as a system’s paging rate or a minidisk’s available space. A data value that exceeds its set limit causes the applicable node ID, value, and value-related information to be highlighted on the panels, thereby warning you of potential system performance or minidisk problems. Press PF2 at any time to obtain the latest data.

System Selection Panel

```
pits
               z/VM SYSWATCH:  System Selection          Line 1 of 2
------------------------------------------------------------------------

To view the data from a monitored system, move the cursor to its node ID
and press Enter.

ATLANTA  BALTIMOR  ENDICOTT  LOSANGEL  MANASSAS  NEWYORK  ORLEANS
PHOENIX  STLOUIS   TUCSON









===>
PF1 Help       PF2 Refresh    PF3 End    PF4 All          PF6 Colors
PF7 Backward   PF8 Forward
```

Figure 83. The System Selection Panel

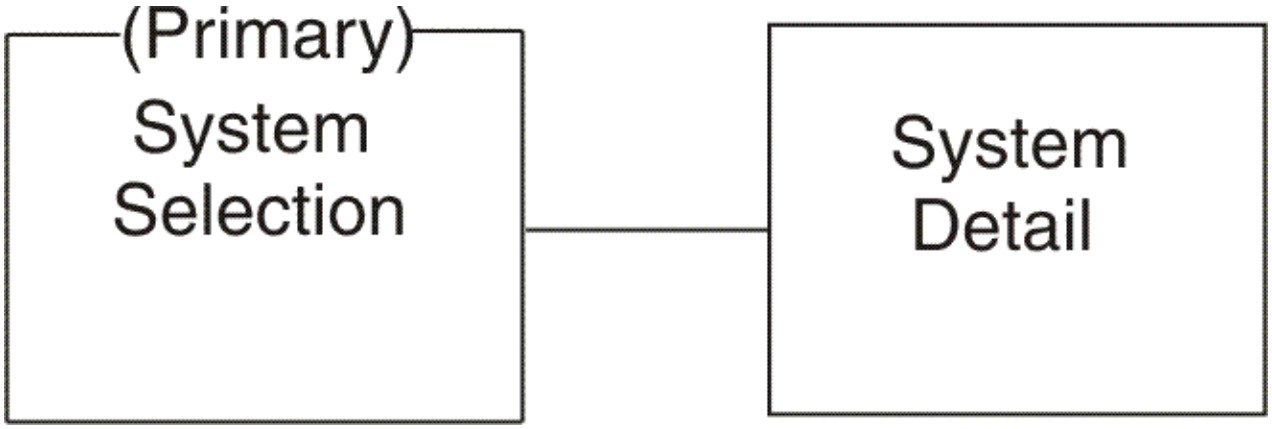
The System Selection panel initially displays all monitored systems currently in **exception mode**. A system is in exception mode when one or more of its variables' values (that is, data returned by an exit routine) exceeds a preset warning level *or* when no data has been received from that system in over 5 minutes. When you press PF4, SYSWATCH displays all the monitored systems, including those not in exception mode. Pressing PF4 again will show only exceptions.

SYSWATCH highlights an exception in one of three colors: WHITE when variable data exceeds warning level 1, YELLOW when variable data exceeds warning level 2, and RED when variable data exceeds warning level 3, or has not been updated within the last 5 minutes. Otherwise, the text color is TURQUOISE. SYSWATCH highlights a node ID in the color of the highest warning level exceeded on that node. You can change these default colors from the Exception Colors panel, which you display by pressing PF6.

If you use a monochrome display, exception node IDs will appear on your screen in a higher intensity of the usual text color. *This will occur regardless of the particular warning level exceeded*. To determine the specific warning level exceeded, select the System Detail panel for the highlighted node ID and compare the data value in exception to the values shown in the three warning level fields.

From the System Selection panel, you select a system for detailed display by moving the cursor under its node ID and pressing Enter. Doing so displays the System Detail panel.

System Detail Panel




```
               z/VM SYSWATCH:  System Detail for LOSANGEL  Line 1 of 20
------------------------------------------------------------------------------

Value Description  Units  Value                 Level 1  Level 2  Level 3
-----------------  -----  --------------------- -------  -------  -------
System Time               14:55:54 PST WEDNESDAY
                          02/01/01
Processor 1 Load   %      081                   75       85       90
Processor 2 Load   %      079                   75       85       90
Expansion Factor          011
Paging             /Sec   0184                  300      400      500

Storage Used       %      032                   50       75       85
Number of Users           221                   250      300      350
Spool Files               1693                  8500     8750     9000
Number of RDR files       1335
Number of PRT files       0328
Number of PUN files       0030
Serial Number             012345
Model Number              2064
Time of Last IPL          01/25/01 05:02:47 PST

===>
PF1 Help       PF2 Refresh    PF3 End      PF6 Colors
PF7 Backward   PF8 Forward
```

Figure 84. The System Detail Panel. This is an example panel showing sample data. Your display should look similar to this one, but your data and data format depends on how you set up SYSWATCH.

The System Detail panel displays all the data SYSWATCH retrieved for a single system. SYSWATCH obtains data using exit routines you provide, or sample exit routines provided, or both. If you use AUDITOR, SVM status will be displayed along with the other data. How all this data appears on the panel is determined by entries in the SYSWATCH THRESHLD file.

Field Descriptions: The System Detail panel shows all the relevant data SYSWATCH obtained for the selected node. These fields are:

Value Description
: This field gives a brief description of the *value*—the AUDITOR or exit routine data shown in the Value field.

Units
: This field shows the units of measure the data in the Value field is expressed in, if applicable. For example, if the value is system paging rate data, the units of measure may appear as "/Sec," for pages *per second*.

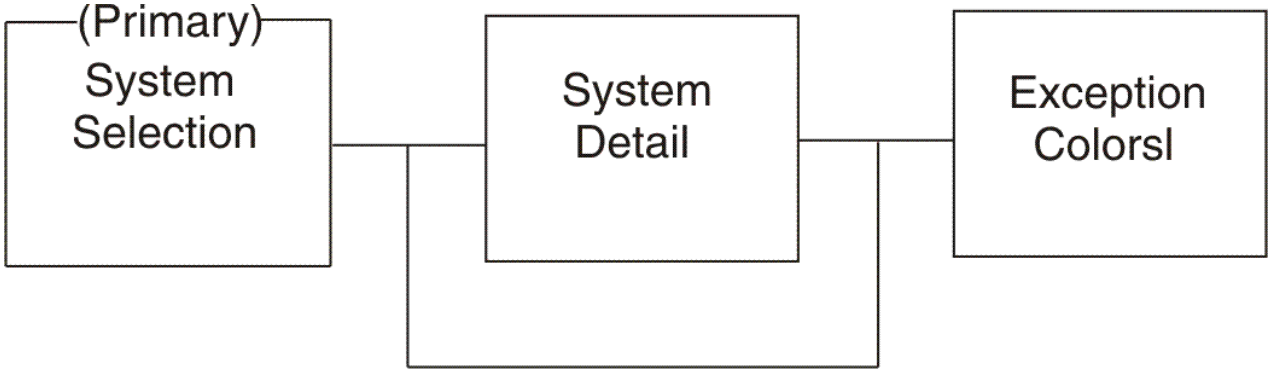
Value
: This field shows the actual system, SVM, or minidisk data retrieved by SYSWATCH. This data is described in the Value Description field.

Level 1
This field shows the first warning level, a limit that, if exceeded by a value, will cause the panel line containing that value to be highlighted. The node ID for the system returning the value will likewise be highlighted on the System Selection panel. Highlighting will be in the Level 1 color named on the Exception Colors panel.

Level 2
This field shows the second warning level. Highlighting will be in the Level 2 color named on the Exception Colors panel. Level 2 highlighting will override Level 1 highlighting.

Level 3
This field shows the third warning level. Highlighting will be in the Level 3 color named on the Exception Colors panel. Level 3 highlighting will override Level 2 or Level 1 highlighting.

Exception Colors Panel




```
               z/VM SYSWATCH:  Exception Colors
------------------------------------------------------------------------

To change the highlight color for any level, type a valid color entry
over the existing color entry and press Enter.

Valid color entries are WHITE, BLUE, RED, YELLOW, PINK, GREEN and
TURQ (turquoise).
------------------------------------------------------------------------


                  Highlighting Level      Text Color
                  --------------------    ----------
                  Normal (no highlight)   TURQ
                  Level 1 highlighting    WHITE
                  Level 2 highlighting    YELLOW
                  Level 3 highlighting    RED



===>
PF1 Help       PF3 End       PF9 Defaults
```

Figure 85. The Exception Colors Panel. This is an example panel showing default text colors. Your panel should look similar when displaying the default colors.

The Exception Colors panel lets you view or change the highlight colors for the three warning levels, and the normal text color for the panel data. You can change any color on the panel by typing over it with a valid color name. To restore SYSWATCH's default colors, press PF9.

You can select this panel from either the System Selection panel or the System Detail panel, by pressing the PF6 key.

Field Descriptions: This panel shows the normal and warning level colors currently in use by SYSWATCH. The fields are:

Highlighting Level
This field lists the categories SYSWATCH uses to classify system node IDs and their related data for highlighting. SYSWATCH categorizes data based on a comparison to the values assigned to fields 6, 7, and 8 in the SYSWATCH THRESHLD file. If you want to check these values, they are displayed in the Level 1, 2, and 3 fields on the System Detail panel.

The node IDs and data for systems whose data exceeds these values are highlighted on the panels, in the text color of the appropriate warning level. The highest warning level exceeded determines the prevailing color. The Exception Colors panel lists the SYSWATCH highlighting levels from lowest to highest.

Text Color
This field shows the color associated with each of the four categories defined in the Highlighting Level field. Valid color entries are `WHITE, BLUE, RED, YELLOW, PINK, GREEN,` and `TURQ` (turquoise). To change a color, type over the existing color entry with any of the valid color entries, and press Enter.

SYSWATCH Panel PF Keys

Each SYSWATCH panel has a number of PF keys available for your use. Here is a list of all the PF keys used by SYSWATCH, and what they do. Each SYSWATCH panel uses some of these available keys:

PF1 Help
Displays help information for this panel.

PF2 Refresh
Displays the latest updated information on the screen.

PF3 End
Returns you to the previous screen. If you are at the primary (System Selection) panel, pressing PF3 ends the SYSWATCH session.

PF4 All
Displays a list of all the systems monitored by SYSWATCH, including systems *not* in exception mode. Press PF4 again to display exceptions only.

PF4 Exceptions
Displays a list of all the systems monitored by SYSWATCH that exceed a warning level set in SYSWATCH THRESHLD or that have sent no data in over 5 minutes. Press PF4 again to display all systems.

PF6 Colors
Displays the Exception Colors panel. Use this panel to check or change text colors.

PF7 Backward
Moves you back one page on the screen.

PF8 Forward
Moves you ahead one page on the screen.

PF9 Defaults
Displays the default (preset) colors in the Text Color field.

Messages and Return Codes

- DMS015E Unknown {CP/CMS} command
- DMS514E Return code *nn* from *command*
- DMS2369I SYSMON is being initialized.
- DMS2370S *Command* is unable to continue because the file *fileid* is missing. [RC=28]
- DMS2371I SYSMON is being shut down.
- DMS2372I *Userid* at *nodeid* has not received data from any systems.
- DMS2373I There are no systems in exception status.
- DMS2374W No data has been received from this system in over 5 minutes.

- DMS2375S A system error was encountered while reading the file *fileid*. [RC=*xx*]
- DMS2376W The file *fileid* could not be found, so *userid* at *nodeid* will be the central site collection id.
- DMS2377W The keyword CENTRAL could not be found in the file *fileid*, so *userid* at *nodeid* will be the central site collection id.
- DMS2378I No user authorizations were specified, so all users will be allowed to access the data.
- DMS2379S No response received from the service machine *userid* at *nodeid*.
- DMS2380S You are not authorized to view this data.
- DMS2381W No data has been received from this system.
- DMS2382I From: VM SYSMON Service Machine*userid* at *nodeid hh:mm:ss*
  on *yy/mm/dd*
  This SYSMON service machine
  has ended abnormally.
  The following error occurred:[RC=xx]

Return codes:

RC
Meaning

0
Normal exit

28
Required file was not found

xx
Return code *xx* from EXECIO, SMSG, MSG, ESTATE, or WAKEUP

Any other return code you receive indicates either a program named in message DMSCYM514E failed, or EXECIO failed. To find the cause of an EXECIO error, see “EXECIO” on page 229.

SYSWATCH: Requirements, Setup, and Use

The SYSWATCH utility collects system performance and other data from centrally-managed systems, and forwards it to the central site. There, system support personnel can view this data by displaying a series of panels. This data can help your support personnel detect problems before users at the monitored systems encounter them. The central site can monitor itself along with the other centrally-managed systems, which themselves can be local or distributed. You can also use SYSWATCH in the single system environment.

The kind of data SYSWATCH collects is determined by exit routines. You can write your own or use the samples provided. If you use the samples, SYSWATCH will obtain and send minidisk information to the central site, along with these system values:

- System serial and model numbers
- Number of RDR, PRT, PUN files
- Number of logged-on users
- Date and time of last IPL
- Percent of processor running time
- Paging rate, percent of pages in storage, expansion factor (average elapsed time ratio for the contention of processor and storage resources)

Besides exit routines, you can set up SYSWATCH to work with AUDITOR. An exit in AUDITOR lets SYSWATCH access SVM data, so SYSWATCH can send AUDITOR's SVM information to the central site, too.

As an authorized user, you can display monitored information using the SYSWATCH panels. You can refresh the screen at any time to get the most current data. You can also set various limits for the data being displayed. If a particular system's data does not match the limits you specify, the system's node ID appears highlighted on the primary panel. From there you can display the detailed data for the highlighted system. All data is displayed, and out-of-limits data is highlighted. If you wish, you can ask SYSWATCH to show you data from any monitored system, highlighted or not.

To display the SYSWATCH data, you must be an authorized user whose virtual machine meets the conditions specified in "Preparing Your System Environment" on page 1380.

Execs That SYSWATCH Uses to Monitor Your Systems

To display SYSWATCH data on your screen, run the following exec on any authorized user's virtual machine:

- SYSWATCH EXEC

To obtain the SYSWATCH data, run the following two execs at the central system and each monitored system, on their own disconnected virtual machines:

- AUDITOR EXEC (if you want SVM data)
- SYSMON EXEC

The SYSWATCH and SYSMON execs work together as part of the SYSWATCH utility. AUDITOR is another CP utility. To run properly, all these execs require input files for instructions.

SYSWATCH EXEC

SYSWATCH EXEC runs on any authorized end-user virtual machine. It requests the data that has been collected at the central system by its counterpart, SYSMON EXEC. It then organizes the data SYSMON returns, and displays it in panels. SYSWATCH EXEC reads two input files:

- SYSWATCH CONTROL, to determine the central system from which to request data
- SYSWATCH THRESHLD, to determine how to display that data

AUDITOR EXEC

AUDITOR EXEC is part of the AUDITOR utility. AUDITOR checks the status of a system's SVMs. AUDITOR tells you the states of your SVMs: which ones are running normally, are logged off, are in a disabled wait state, have failed tests made by exit routines, and so on. You set up AUDITOR to run automatically at each system on its own virtual machine. Using the SENDSTAT exit in the AUDITOR OPTIONS file, you can forward SVM data to SYSMON at the central site, where it can be displayed by SYSWATCH.

See "Using AUDITOR with SYSWATCH" on page 1379 for a description of how to use the SENDSTAT exit. See "AUDITOR: Requirements, Setup, and Use" on page 1302 for a detailed description of AUDITOR and its required input files.

SYSMON EXEC

SYSMON EXEC runs at each monitored system, where it invokes exit routines to generate data for SYSWATCH. SYSMON EXEC reads three input files:

- At each monitored system (including the central system) it reads:
  - SYSWATCH EXITS, to determine what exit routines to run, and how often to run them
  - SYSWATCH CONTROL, to determine the central system to receive the data, and the users to be notified should SYSMON fail
- At the central system only, it reads:
  - SYSWATCH THRESHLD, to determine what each system's warning levels are for the collected data
  - SYSWATCH CONTROL, to determine the users authorized to request the data, and to identify those systems whose data must be kept current

SYSMON at each monitored system gets the data and sends it to the central system, where it is collected. SYSMON at the central system responds to queries from authorized SYSWATCH users, sending them their requested data for display.

SYSWATCH Input Files

Before running SYSWATCH (see "Beginning System Monitoring" on page 1381), you need to prepare the following input files:

- SYSWATCH CONTROL
- SYSWATCH EXITS
- SYSWATCH THRESHLD

SYSWATCH CONTROL File

The monitored systems use the SYSWATCH CONTROL file to identify these IDs:

- The central system node ID and user ID collecting the data
- The user IDs authorized to request the data (at the central system only)
- The user IDs to be notified should SYSMON fail

The central monitoring system uses the SYSWATCH CONTROL file to identify those monitored systems whose data it must keep current. Copies of SYSWATCH CONTROL should reside at each monitored system.

Both the SYSMON and SYSWATCH EXECs must have access to SYSWATCH CONTROL. If you use the sample SENDSTAT exit routine to get AUDITOR data, the AUDITOR EXEC must have access to it as well.

CONTROL File Format

Each control statement consists of a keyword and a value, formatted as:

```
KEYWORD value
```

For example,

```
CENTRAL  SYSMON AT CENTRAL
```

is a control statement telling SYSMON the user ID and node ID at the central site where SYSWATCH data is to be sent and collected. Records that do not begin with the keywords CENTRAL, AUTHORIZE, REMOTE, or ADMIN are ignored, and lines that begin with an asterisk (*) or slash asterisk (/*) are treated as comments.

The following table describes the keywords and their values.

Table 68. SYSWATCH CONTROL Keywords

Keyword	Description
`CENTRAL` *userid* `AT` *nodeid*	Specifies the user ID and node ID of the central system virtual machine collecting the SYSWATCH data.
`AUTHORIZE` *userid* `AT` *nodeid*	Specifies a user ID authorized to request SYSWATCH data. Include one AUTHORIZE statement for each SYSWATCH user. Specify `ALL` instead of *userid* `AT` *nodeid* to indicate that any user may request SYSWATCH data. The default value is `ALL`.
`REMOTE` *userid* `AT` *nodeid*	Specifies a user ID running SYSMON EXEC. Include one REMOTE statement for each user ID running SYSMON. Whenever the virtual machine running SYSMON at the central system is started or restarted, SYSMON does the following: For each user ID represented by a REMOTE statement, SYSMON 1. Forces their exit routines to run immediately, and 2. Refreshes their data. Any user ID that fails to send data will have its node ID highlighted on the System Selection panel. However—user IDs running SYSMON but *not* represented by a REMOTE statement *are not checked* for data. Should one of these user IDs go down during a restart of SYSMON at the central site, the failure may be difficult to detect because its node ID simply would not appear when the System Selection panel returns. Therefore, it is recommended you include a REMOTE statement for each user ID running SYSMON. The REMOTE statement is optional.
`ADMIN` *userid* `AT` *nodeid*	Specifies a user ID authorized to receive a SYSMON FAILURE file, in the event the virtual machine running SYSMON ends processing abnormally. This file includes an error message describing the failure. It is sent to the reader of the specified user ID, in addition to any message displayed at the console of the virtual machine running SYSMON. Include one ADMIN statement for each user you want notified, should SYSMON fail. The ADMIN statement is optional.

Sample CONTROL File

Use the following sample SYSWATCH CONTROL file as a guide when tailoring the SYSWATCH CONTROL file provided.

```
*************************************************************************
*                                                                       *
*                           SYSWATCH CONTROL                            *
*                                                                       *
*************************************************************************
*                                                                       *
*  This file specifies control information for the SYSWATCH utility.    *
*                                                                       *
*************************************************************************

* Identify the user ID and node of the central system collecting the
* data.
*
CENTRAL  SYSMON AT CENTRAL

* Identify the user IDs authorized to display SYSWATCH data.  Specify
* AUTHORIZE ALL to let any user ID display the data.
*
* IMPORTANT NOTE:  If you do NOT use an AUTHORIZE statement, it is the
* same as having an AUTHORIZE ALL statement.
*
AUTHORIZE  OPERVIEW AT CENTRAL
AUTHORIZE  MAINT    AT CENTRAL
AUTHORIZE  BOSS     AT CENTRAL

* Identify the user IDs running SYSMON EXEC.  This statement is optional,
* but it does tell SYSWATCH to highlight systems that do not send data
* (indicating possible network problems), and to refresh all data when
* the central monitoring system is started.
*
REMOTE  SYSMON AT ATLANTA
REMOTE  SYSMON AT BALTIMOR
REMOTE  SYSMON AT ENDICOTT
REMOTE  SYSMON AT LOSANGEL
REMOTE  SYSMON AT MANASSAS
REMOTE  SYSMON AT NEWYORK
REMOTE  SYSMON AT ORLEANS
REMOTE  SYSMON AT PHOENIX
REMOTE  SYSMON AT STLOUIS
REMOTE  SYSMON AT TUCSON

* Identify the administrator user IDs.  These user IDs will be notified
* whenever SYSMON terminates abnormally.
*
ADMIN  MAINT AT CENTRAL
```

Figure 86. Sample SYSWATCH CONTROL File

SYSWATCH EXITS File

Each monitored system uses the SYSWATCH EXITS file to identify the exit routines to be run, and the time intervals at which to run them. A copy should reside at each system, on any minidisk or directory accessed by the virtual machine running SYSMON. Copies of the file may differ if you want to run different exit routines at different systems. SYSMON EXEC must have access to this file.

EXITS File Format

The SYSWATCH EXITS file contains a series of exit statements, each naming an exit routine to use to retrieve data, and a time interval for running the routine. For each exit routine prepare one statement consisting of two values, in the format:

```
exit_routine +hh:mm:s
```

The first value is the name of an exit routine. The second value is the run interval. Any information following the second value is treated as a comment. For example,

```
QTIME  +00:03:0  Get the system time using QUERY TIME
```

is an exit statement telling SYSWATCH to run the QTIME exit routine every three minutes to obtain the date and time from the system. Blank lines are ignored, and lines beginning with an asterisk (*) or slash asterisk (/*) are treated as comments.

The following table describes the exit statement values.

Table 69. SYSWATCH EXITS Values

Value	Description
exit_routine	Specifies the name of an exec to be run as an exit routine. Data retrieved by this exit routine will be forwarded to the central system for display.
+hh:mm:s	Specifies how often the exit routine will be run. Specify the interval in *+hh* (00–23) hours, *mm* (00–59) minutes, and *s* (0–5) seconds, where seconds is in multiples of ten. At each monitored system, run at least one exit routine at a time interval of 5 minutes or less. SYSWATCH uses this time interval to call your attention to possible system networking problems. It does so by highlighting on the System Selection panel the node ID of any system that has not returned data in 5 minutes. If all exit routines on one system run at intervals longer than 5 minutes, that system will always be highlighted. But by running just one routine at a shorter interval, only those systems that may be functioning improperly will be highlighted.

Sample EXITS File

Use the following sample SYSWATCH EXITS file as a guide when tailoring the SYSWATCH EXITS file provided.

```
*************************************************************************
*                                                                       *
*                            SYSWATCH EXITS                             *
*                                                                       *
*************************************************************************
*                                                                       *
*  This file specifies the exit routines used by the SYSWATCH utility,  *
*  and the time intervals at which they run.                            *
*                                                                       *
*************************************************************************

QTIME    +00:03:0  Get the system time using QUERY TIME
CPIND    +00:03:0  Get the CPU load, exp factor, and so on, using CP IND
QFILES   +00:15:0  Get the number of spool files using QUERY FILES
QUSERS   +00:03:0  Get the number of logged-on users using QUERY USERS
LASTIPL  +23:59:0  Get the last IPL time and date using QUERY CPLEVEL
QCPUID   +23:59:0  Get the CPU serial and model numbers using QUERY CPUID
QDISK    +06:00:0  Get the minidisk percent full data using QUERY DISK
```

Figure 87. Sample SYSWATCH EXITS File

SYSWATCH THRESHLD File

The SYSWATCH THRESHLD file formats the exit routine data for display on the System Detail panel, as shown in "System Detail Panel" on page 1365. For each type of data retrieved, SYSWATCH THRESHLD describes the data, the sequence (row) in which it will appear on the panel, its units of measure (percent as %, per second as /Sec, and so on), the limits or warning levels you set for the data (if any), and the system or systems to which these limits apply.

SYSWATCH THRESHLD also records the variable name associated with the exit routine's data; this is the name used by the exit routine when it queues the data on the stack.

A copy of SYSWATCH THRESHLD should reside at the central system on a minidisk or directory accessed by the virtual machine running SYSMON. SYSWATCH THRESHLD is required by SYSWATCH EXEC, and by SYSMON EXEC at the central system. Additional copies should be kept on the virtual machines of authorized SYSWATCH users, if they are on different systems or if they want to tailor their own data descriptions, warning levels, or screen layout.

THRESHLD File Format

Threshold statements consist of ordered, column-dependent fields, each field having a specified value. Each statement describes how the data related to one particular exit routine will be displayed. Blank lines are ignored, and lines beginning with an asterisk (*) or slash asterisk (/*) are treated as comments.

The following table describes the SYSWATCH THRESHLD fields.

Table 70. SYSWATCH THRESHLD Fields

Field	Columns	Value	Description
1	1 to 8	*system*	Specifies the system(s) to which this threshold statement applies. Specify ALL to indicate *all* systems. Specify model number *nnnn* to indicate all systems with a particular four-digit processor model number (digits 9–12 from the ID number returned by the CP QUERY CPUID command); for example, you might specify 2064. This overrides any ALL setting already set for the same variable. Specify node ID *nodeid* to indicate the node identifier for a particular system. This overrides any ALL or *nnnn* settings already set for the same variable.
2	10 to 13	*row*	Specifies the row, or line number, where data related to the variable named in field 3 will be displayed on the System Detail panel. For example, if you specified row as 5, the variable would be displayed as the fifth data line on the panel. Rows 1 through 15 will appear on the first panel; scroll to the next panel to see higher row numbers. **Note:** In the sample SYSWATCH THRESHLD file shown in Figure 88 on page 1377, the same row is sometimes used for more than one variable. This can be done when you know a system cannot report both variables that use the same row number. For example, a system would not report both MPLoad and Proc1Load.
3	15 to 28	*variable*	Specifies the variable name. This is the unique identifier given to a type of data by the exit routine that retrieves it. This variable name ties to the System Detail panel the data generated by the exit routine. Therefore the variable name specified here and the one specified in the exit routine must match exactly. See “Understanding the SYSWATCH Exit Routines” on page 1379 for more information. Specify the variable name as an alphanumeric character string with up to 14 characters and no embedded blanks. **Note:** If you are using AUDITOR to forward SVM status from any system, specify the monitored SVM’s user ID in field 3. This identifies the SVM whose status will be displayed.
4	30 to 48	*description*	Gives a brief description of the data assigned the variable name in field 3. This description will appear on the System Detail panel, to identify the data for the panel user.
5	50 to 55	*units*	Specifies the units of measure the data is expressed in, whenever applicable. For example, if the data retrieved is the system paging rate, you might express units of measure in pages *per second* (/Sec). This unit's information accompanies its related data on the System Detail panel. Use of field 5 is optional.

Table 70. SYSWATCH THRESHLD Fields (continued)

Field	Columns	Value	Description
6	57 to 63	*level1*	Specifies the Level 1 warning level—a value you set as a limit for any data associated with the variable named in field 3. If the data exceeds this limit, SYSWATCH will highlight on the primary panel (which just displays node IDs) the node ID of the system whose data is in exception (see "System Selection Panel" on page 1365). The data itself will be highlighted on the System Detail panel. **Note:** If you have several end-user virtual machines displaying SYSWATCH data, and they use versions of SYSWATCH THRESHLD that specify warning levels different from those at your central site: The warning levels specified at the central site will determine node ID highlighting on the System Selection panel. The warning levels specified at each end-user system will determine the data line's highlighting on the System Detail panel. With SYSWATCH you can associate three warning levels with each variable's data, Level 1 being the first of three. Each warning level has a different highlight color. Any data below Level 1 is considered normal, and is not highlighted. Should the data exceed one of the three warning levels, all highlighted information will appear in the color for the highest warning level exceeded. You can use SYSWATCH's default highlight colors or, using the Exception Colors panel, change the colors for each warning level to the ones you want. Use of field 6 is optional.
7	65 to 71	*level2*	Specifies the Level 2 warning level value. Use of field 7 is optional.
8	73 to 80	*level3*	Specifies the Level 3 warning level value. Use of field 8 is optional.

Sample THRESHLD File

Use the following sample SYSWATCH THRESHLD file as a guide when tailoring the SYSWATCH THRESHLD file provided.

```
*************************************************************************
*                                                                       *
*                            SYSWATCH THRESHLD                          *
*                                                                       *
*************************************************************************
*                                                                       *
*This file specifies panel layout, variable names, and warning level values *
*for the SYSWATCH utility.                                              *
*                                                                       *
*************************************************************************

*---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
*       |    |             |                |      |       |       |       |
*System |Row |Variable     |Variable        |Units |Level1 |Level2 |Level3 |
*       |    |Name         |Description     |      |       |       |       |
*       |    |             |                |      |       |       |       |
ALL      1    SysTime       System Time
ALL      2    MPLoad        MP Processor Load   %          75      85      90
ALL      3    APLoad        AP Processor Load   %          75      85      90
ALL      2    Proc1Load     Processor 1 Load    %          75      85      90
ALL      3    Proc2Load     Processor 2 Load    %          75      85      90
ALL      2    AvgProc       Processor Load (z/VM)%         75      85      90
ALL      4    ExpFactor     Expansion Factor
ALL      5    Paging        Paging              /Sec       300     400     500
2064     5    Paging        Paging              /Sec       150     200     300
ALL      6    Swapping      Swapping (HPO)      /Sec       400     500     600
ALL      7    Storage       Storage Used        %           50      75      85
ALL      8    NumberUsers   Number of Users               750     900    1000
LOSANGEL 8    NumberUsers   Number of Users               250     300     350
ALL      9    SpoolFiles    Spool Files                 8500    8750    9000
ALL      10   RdrFiles      Number of RDR files
ALL      11   PrtFiles      Number of PRT files
ALL      12   PunFiles      Number of PUN files
ALL      13   SerialNumber  Serial Number
ALL      14   ModelNumber   Model Number
ALL      15   IPLTime       Time of Last IPL
ALL      16   CMSAPPL       CMSAPPL SVM Status            2       3       4
ALL      17   CMSAPPL.191   CMSAPPL 191 Usage   % Full   85      90      95
ALL      18   CMSSFS        CMSSFS  SVM Status            2       3       4
ALL      19   CMSSERV       CMSSERV SVM Status            2       3       4
ALL      20   SYSMON.191    SYSMON  191 Usage   % Full   85      90      95
```

Figure 88. Sample SYSWATCH THRESHLD File

Sample Files That Support SYSWATCH

The following sample files are provided to support SYSWATCH:

- PROFILE EXEC
- Exit routines
- QDISK CONTROL

Sample PROFILE EXEC

A sample PROFILE EXEC is provided on the SYSMON 191 minidisk. This exec uses the following line to start the SYSMON EXEC:

```
EXEC SYSMON
```

This PROFILE EXEC must reside on the A-disk or directory of each virtual machine running SYSMON.

Sample Exit Routines

To monitor your systems, optional exit routines you can run at each system are provided. Use these routines with SYSWATCH:

Table 71. Sample SYSWATCH Exit Routines Provided	
Exit Routine	**Description**
CPIND EXEC	Gets the processor load, expansion factor, processor storage used, paging rate, and swapping rate using CP INDICATE LOAD
LASTIPL EXEC	Gets the last IPL time and date using QUERY CPLEVEL (the date format set for your virtual machine is used)
QCPUID EXEC	Gets the processor serial and model numbers using QUERY CPUID
QDISK EXEC	Gets the minidisk percent full data using QUERY DISK
QFILES EXEC	Gets the number of spool files using QUERY FILES
QTIME EXEC	Gets the system time using QUERY TIME
QUSERS EXEC	Gets the number of logged-on users using QUERY USERS

Use this routine if you use AUDITOR with SYSWATCH:

Table 72. Sample AUDITOR Exit Routine Provided	
Exit Routine	**Description**
SENDSTAT EXEC	Forwards SVM status information to SYSMON at the central system

To use any of these or your own exit routines, make sure the SYSWATCH EXITS file and SYSWATCH THRESHLD file have been set up to handle them (see "Understanding the SYSWATCH Exit Routines" on page 1379).

Each sample exit routine is self-contained, except for QDISK EXEC which requires a QDISK CONTROL file.

QDISK CONTROL File

Before running the sample QDISK EXEC to get minidisk information (see "Beginning System Monitoring" on page 1381), you will need to prepare a QDISK CONTROL file. This file provides SYSWATCH with the information it needs to locate the minidisks you want to check.

Each control statement consists of either two or three values, in the format:

userid owner_address [user_address]

The first value is the user ID of the minidisk. The second value is the address of the minidisk. The third, optional value is a SYSMON address you provide if the minidisk is linked to SYSMON. For example,

```
AUDITOR 191 321
```

is a control statement telling SYSWATCH to check the AUDITOR user ID's A-disk, 0191, which is linked to SYSMON at address 0321. Blank lines are ignored, and lines beginning with an asterisk (*) or slash asterisk (/*) are treated as comments.

Here is a description of the values included in the QDISK CONTROL file:

Table 73. QDISK CONTROL Values	
Value	**Description**
userid	Specifies the user ID of the minidisk owner.
owner_address	Specifies the owner address of the minidisk.
user_address	Specifies the address in SYSMON's directory entry that links the minidisk to SYSMON, if applicable. If you specify no *user_address*, SYSMON must be able to obtain a read link to this minidisk without a password (that is, either its read password is ALL, or SYSMON has read authority).

Use the following sample QDISK CONTROL file as a guide when tailoring the QDISK CONTROL file provided.

```
*************************************************************************
*                                                                       *
*                             QDISK CONTROL                             *
*                                                                       *
*************************************************************************
*                                                                       *
*  This file specifies the minidisks to be queried by QDISK EXEC from   *
*  the QDISK exit of SYSMON EXEC, a part of the SYSWATCH utility.       *
*                                                                       *
*************************************************************************

* The format for each statement is:  userid owner_address user_address
* where user_address is optional.

* These disks are linked in the user's directory, or owned by the user:
*
SYSMON   191 191

* These disks are not linked by the user, but the user can link to them
* (SYSMON has read authority to the disk, or its read password is ALL).
*
CMSAPPL  191
```

Figure 89. Sample QDISK CONTROL File

Understanding the SYSWATCH Exit Routines

PI

SYSWATCH runs exit routines to get the data it displays on its panels; it provides you with sample exit routines that, with minor modifications, are “ready-to-use.” Look at the samples to see how they work. They were written in REXX and set up using the steps that follow. They may need to be tailored to your installation. To add your own exit routines to SYSWATCH, you need to follow the same three steps:

1. Create the exec to the following specifications. The exec must queue its data on the stack upon exit, so SYSMON EXEC can pick it up. Each line queued should be in the format:

   variable_name data

   where *variable_name* is a tag used by SYSMON EXEC when retrieving the queued data from the stack. The variable name can not be longer than 14 alphanumeric characters. *Data* is the actual data collected, which could be null. (Null data can be useful if you write an exec to return error conditions, such that it returns a null value when no errors are found.)
2. Add a statement to SYSWATCH THRESHLD for each variable queued by the exit routine. See “SYSWATCH THRESHLD File” on page 1374 for information on how to do so. The variable name in field 3 (columns 15 to 28) of SYSWATCH THRESHLD must match exactly the variable name queued by the new exit routine.
3. Add an exit statement for the new routine to SYSWATCH EXITS. Be sure the statement is in the SYSWATCH EXITS file on each system where you want the exit routine to run. See “SYSWATCH EXITS File” on page 1373 for information on writing exit statements. Then install the routine itself at the applicable systems, on any minidisk or directory accessed by the virtual machine running SYSMON.

Using AUDITOR with SYSWATCH

Using the EXIT SENDSTAT Option to Send SVM Data to the Central Site: SYSWATCH can forward SVM status from a monitored system to the central site by means of the SENDSTAT exit in AUDITOR. A sample exit routine, SENDSTAT EXEC, is provided for use with this exit. To use this sample, add the line

```
EXIT SENDSTAT SENDSTAT
```

to the AUDITOR OPTIONS file. The SENDSTAT exit routine is executed anytime an SVM’s status is checked. It gets the status information, formats it for SYSWATCH, and sends it to the central monitoring system named in SYSWATCH CONTROL. To ensure you can display the data once it is received, verify you have added a statement to SYSWATCH THRESHLD for each monitored SVM. Use the SVM’s user ID as the variable name in field 3 (see Table 70 on page 1375, and Figure 88 on page 1377).

To use the SENDSTAT exit with your own exit routine, *exec_name*, include an option statement in the format

`EXIT SENDSTAT` *exec_name*

in AUDITOR OPTIONS. The exit routine is passed a string in the format:

userid state

where *userid* is the service virtual machine monitored by AUDITOR, and *state* is the corresponding status of that machine (UP, DOWN, SERVICE, CYCLE, RECYCLED, IGNORE, FAILURE, PD, or OFF). The SENDSTAT exit takes this string, formats it for SYSWATCH, and sends it to the central monitoring system named in SYSWATCH CONTROL.

See "AUDITOR" on page 1297 for details on AUDITOR and its input files.

Setting the Warning Levels for SVMs:

SENDSTAT EXEC associates a number with each of the AUDITOR states. The numbers are:

1=UP
2=DOWN, SERVICE, CYCLE, RECYCLED, IGNORE
3=FAILURE, PD
4=OFF

You can set meaningful warning levels for SVMs by using the above numbers for the Level 1, 2, and 3 values in the SYSWATCH THRESHLD file.

For example, say that for a monitored SVM, CMSAPPL at LOSANGEL, you specify 2 for Level 1, 3 for Level 2, and 4 for Level 3 in SYSWATCH THRESHLD, just like in Figure 88 on page 1377. Then the System Detail panel for node LOSANGEL will display in its Value field for CMSAPPL:

1=UP (in the normal text color), if CMSAPPL's state is UP
2=DOWN, SERVICE, CYCLE, RECYCLED, IGNORE (in the Level 1 color), if CMSAPPL's state is DOWN, SERVICE, CYCLE, RECYCLED, or IGNORE.
3=FAILURE, PD (in the Level 2 color), if CMSAPPL's state is FAILURE or PD
4=OFF (in the Level 3 color), if CMSAPPL's state is OFF.

PI end

Preparing Your System Environment

Each virtual machine running SYSMON requires:

- A R/W minidisk or SFS directory with space enough to hold work files, control files, and exit routines. At least 5 cylinders of 3380 DASD (or the equivalent) are required.
- Read access to the MAINT 193 minidisk, which contains the SYSMON EXEC.
- Access to the exit routines listed in your SYSWATCH EXITS file.
- 5 MB of virtual storage.
- CP user privilege classes sufficient to run the exit routines. If the sample exit routines are used, privilege classes D and G are required. Privilege class E is required when monitoring CP INDICATE LOAD data.

Each virtual machine running SYSWATCH requires:

- Read access to the MAINT 193 minidisk, which contains the SYSWATCH EXEC
- Copies of the SYSWATCH CONTROL and SYSWATCH THRESHLD files on a minidisk or directory accessed by SYSMON
- 5 MB of virtual storage

Beginning System Monitoring

This section describes how to tailor SYSWATCH's files. If you are working with centrally-managed systems, you will be setting up your central monitoring system first. If you are working with a single system, you will be setting that up. In either case, follow these steps:

STEP 1—Setting Up a Virtual Machine to Run SYSMON

1. Ensure you have a virtual machine set up to run the SYSMON EXEC using the criteria specified in "Preparing Your System Environment" on page 1380. Be sure any values you specify meet the criteria described in Table 73 on page 1378.
2. Ensure SYSMON's PROFILE EXEC is set up as specified in "Sample PROFILE EXEC" on page 1377.

STEP 2—Setting Up the SYSWATCH CONTROL File

1. Tailor the SYSWATCH CONTROL file on SYSMON 0191, using Table 68 on page 1372.
2. If you plan to retrieve SVM status and AUDITOR is already set up at your system, copy the tailored SYSWATCH CONTROL file to the A-disk or directory of the AUDITOR virtual machine. In this case, the virtual machines running SYSMON and AUDITOR will have identical SYSWATCH CONTROL files.

STEP 3—Setting Up the SYSWATCH EXITS File

1. Tailor the QDISK CONTROL file on SYSMON 0191, if required. Ensure any values you specify meet the criteria described in Table 73 on page 1378.
2. If you create your own exit routines, copy them to the A-disk or directory of the virtual machine running SYSMON.
3. If you want to:
   - Add or delete exit statements from SYSWATCH EXITS, or
   - Change an exit routine's time interval,

   tailor SYSWATCH EXITS on SYSMON 0191 accordingly.

STEP 4—Setting Up or Modifying AUDITOR

If you want to use AUDITOR to obtain SVM data for display, do the following:

1. Set up AUDITOR, if it has not been done. See "AUDITOR" on page 1297 for details.
2. Add the line

   ```
   EXIT SENDSTAT SENDSTAT
   ```

   to the AUDITOR OPTIONS file, on the A-disk or directory of the AUDITOR virtual machine. (If you are using your own SENDSTAT exit routine, add the line

   ```
   EXIT SENDSTAT exec_name
   ```

   to the AUDITOR OPTIONS file, substituting the file name of your exec for *exec_name*.)
3. If you have not already done so, copy your tailored SYSWATCH CONTROL file to the A-disk or directory of the AUDITOR virtual machine.
4. Start or restart AUDITOR. See "AUDITOR" on page 1297 if you need to know how.

STEP 5—Setting Up the SYSWATCH THRESHLD File

1. Tailor the SYSWATCH THRESHLD file on SYSMON 0191, if necessary. For example, tailor this file if you want to:
   - Add statements for your own exit routines
   - Add statements to track SVMs monitored by AUDITOR

- Rearrange rows on the System Detail panel
- Set your own warning level values

STEP 6—Starting SYSMON EXEC

Begin system monitoring by starting SYSMON EXEC at the virtual machine running SYSMON:

1. Verify SYSMON's PROFILE EXEC is set up as specified in "Sample PROFILE EXEC" on page 1377.
2. If you want SYSMON to start automatically whenever the system is started, add an entry for the virtual machine running SYSMON to AUTOLOG1's PROFILE EXEC. See *z/VM: CP Planning and Administration* for information on how to use AUTOLOG1. Otherwise, log on to the virtual machine running SYSMON and enter the command

   ```
   profile
   ```

   on the command line. Then enter the command

   ```
   #cp disc
   ```

   to disconnect the virtual machine, which will continue running SYSMON.
3. If you are installing a single system, skip the next section and go to "STEP 8—Displaying SYSWATCH Panels" on page 1382.

STEP 7—Setting Up Centrally-Managed Systems

If you are working with centrally-managed systems, you are now finished setting up your central system. But before going to the next step, you need to do the following *for each system you want to monitor:*

1. Repeat "STEP 1—Setting Up a Virtual Machine to Run SYSMON" on page 1381.
2. Copy your central system version of SYSWATCH CONTROL, QDISK CONTROL, and SYSWATCH EXITS
   - *From:* The A-disk or directory of the virtual machine running SYSMON at the central system
   - *To:* The A-disk or directory of the virtual machine running SYSMON at the monitored system.
3. Tailor QDISK CONTROL and SYSWATCH EXITS as necessary. You do not need to change SYSWATCH CONTROL. If you want standardized systems, do not change any of these files.
4. If you created your own exit routines, copy them to the A-disk or directory of the virtual machine running SYSMON.
5. Repeat "STEP 4—Setting Up or Modifying AUDITOR" on page 1381.
6. Repeat the setup and start-up portions (substeps 1 and 2 only) of "STEP 6—Starting SYSMON EXEC" on page 1382.

When you have completed the above six steps for each system you want to monitor, go to "STEP 8—Displaying SYSWATCH Panels" on page 1382.

STEP 8—Displaying SYSWATCH Panels

To display the SYSWATCH panels:

1. Verify the virtual machine where you want to run SYSWATCH EXEC meets the criteria specified in "Preparing Your System Environment" on page 1380.
2. **If you are working with centrally-managed systems** and you want to run SYSWATCH on them, verify the end-user virtual machines you will use to display panels also meet the criteria specified in "Preparing Your System Environment" on page 1380.
3. To start SYSWATCH, log on to a virtual machine running SYSWATCH, and enter the command

   ```
   syswatch
   ```

   on the command line.

4. See “Using the SYSWATCH Panels” on page 1364.

Tailoring Your Input Files

As your system workload and circumstances change, you may need to change your SYSWATCH input files. Whenever you change SYSWATCH CONTROL, SYSWATCH THRESHLD, or SYSWATCH EXITS, you will need to **cycle**—that is, stop and restart—SYSMON EXEC at the system being changed. This cycling of SYSMON EXEC allows the exec to collect new data under the changed conditions.

Here is a table that gives some general information about tailoring your input files.

Table 74. Tailoring SYSWATCH Input Files

Input File to be Tailored	Tailor This File To:	Tailor This File and Cycle SYSMON At:	Other Considerations
SYSWATCH CONTROL	Add a monitored system	The central system	Set up the new monitored system as explained in “STEP 7—Setting Up Centrally-Managed Systems” on page 1382
	Delete a monitored system	The central system	—
	Grant or revoke user authority	The central system	—
	Add or delete an administrator	Each applicable monitored system	—
	Change the user ID or node ID of central system virtual machine collecting the SYSWATCH data	All monitored systems	Set up the new central system, as explained in “Beginning System Monitoring” on page 1381
SYSWATCH THRESHLD	Change warning levels	The central system	Update any additional copies of SYSWATCH THRESHLD you have made
	Reformat the System Detail panel	The central system	Update any additional copies of SYSWATCH THRESHLD you have made
	Add a threshold statement for a new exit routine or SVM	The central system	Update any additional copies of SYSWATCH THRESHLD you have made. Add, for any new routine, an exit statement to SYSWATCH EXITS at each system running the new routine
	Remove a threshold statement for a withdrawn exit routine or SVM	The central system	Update any additional copies of SYSWATCH THRESHLD you have made. Remove, for any unused routine, its exit statement from SYSWATCH EXITS at each system no longer running the routine
SYSWATCH EXITS	Add an exit statement for a new exit routine	Each applicable monitored system	Add a threshold statement for the new routine to SYSWATCH THRESHLD at the central system
	Remove an exit statement for an unused exit routine	Each applicable monitored system	Remove the related threshold statement from SYSWATCH THRESHLD at the central system
	Change an exit routine’s time interval	Each applicable monitored system	Verify one exit routine at this system still runs in intervals under five minutes

Error Checking in SYSWATCH

SYSWATCH performs its own error checking. It issues error messages, as well as informational and progress messages.

SYSMON EXEC reports errors it finds in the virtual machine running SYSMON to the console of that machine. Severe errors, such as failure to find a required control file, cause SYSMON EXEC to end processing and stop. When this happens, a file called SYSMON FAILURE is sent to the reader of any user

ID named in an ADMIN statement in the SYSWATCH CONTROL file (see Table 68 on page 1372). The SYSMON FAILURE file includes a SYSMON error message describing the failure.

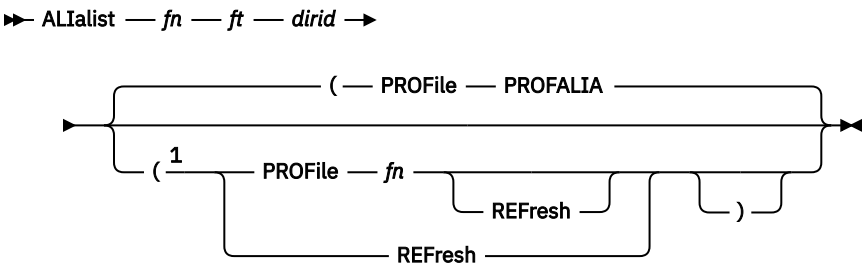
SYSWATCH EXEC reports errors it finds during data display to the SYSWATCH panels, unless they occur during start-up. It reports errors occurring during start-up to the console.

Chapter 4. Special Commands Used Within Other Commands

There are four commands you can use to work with the lists displayed by certain commands. These commands cannot be entered from the command line; they can *only* be used in the command environments listed below. They are:

Command	Can be Issued In
ALIALIST	FILELIST
AUTHLIST	DIRLIST FILELIST
DISCARD	DIRLIST FILELIST MACLIST PEEK RDRLIST
EXECUTE	CSLLIST CSLMAP DIRLIST FILELIST MACLIST RDRLIST VMLINK

ALIALIST



Notes:

[1] You can enter Options in any order between the parentheses.

Authorization

General User

Purpose

Use the ALIALIST command only in the FILELIST environment. PF9 is assigned to ALIALIST on the FILELIST screens for the SHARE and SEARCH option.

The alias information 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 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 another user owns, the file names of the aliases 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, write, DIRREAD, or DIRWRITE 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 because the base file no longer exists.

The information is placed in an XEDIT file named *userid* ALIALIST A0.

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.

Options

PROFile *fn*
specifies the name of an XEDIT macro to be executed when XEDIT is invoked by the ALIALIST command. If not specified, the default macro PROFALIA XEDIT is invoked. For more information on the PROFALIA macro, see Usage Note “3” on page 1387, "Default PF Key Settings for ALIALIST."

REFresh
updates the display of alias information with any changes made to alias information since the last update.

Usage Notes

1. For more information on how to customize this command see Appendix A, “Customizing Profiles for CMS Productivity Aids,” on page 1419.
2. 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.
3. Default PF Key Settings for ALIALIST

   Entering the ALIALIST command executes the PROFALIA XEDIT macro. It sets the keys to these values (the ALIALIST command is only valid for files in SFS directories):

Key	Setting	Action
PF1	Help	Displays the ALIALIST command description.
PF2	Refresh	Updates the list to display new or deleted alias information.
PF3	Return	Return to the FILELIST screen.
PF4	Sort (type)	Sort the list by file type, file name.
PF5	Sort (name)	Sort the list of files by file name, file mode.
PF6	Sort (dir)	Sort the list by directory name, file name, file type.
PF7	Backward	Scroll backward one screen.
PF8	Forward	Scroll forward one screen.
PF9	S(user)	Sort the list by user ID.
PF10		Unassigned.
PF11		Unassigned.
PF12		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 these PF keys, the PROFALIA XEDIT macro sets synonyms 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.

Examples

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.

```
SMITH   ALIALIST    A0  V 190  Trunc=190 Size=4   Line=1 Col=1 Alt=0
Base file = CMSFILES SCRIPT FPOOL1:SMITH.GOODIES.FOOD
Userid   Num Filename Filetype Directory
SMITH      1 CMSFILES SCRIPT   .SALES
JONES      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 90. Sample ALIALIST Screen

For more examples on using the ALIALIST command, see *z/VM: CMS User's Guide*.

Messages and Return Codes

- DMS651E ALIALIST must be issued from FILELIST [RC=40]
- DMS653E Error executing QUERY ALIAS, rc=*nn* [RC=*nn*]
- DMS1230E ALIALIST is invalid for minidisk file [RC=0]
- DMS1231E ALIALIST is invalid on a directory [RC=0]
- DMS1233E Invalid use of REFRESH option [RC=40]
- DMS9059E No aliases for this base file [RC=0]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	“Command Syntax Error Messages” on page 1411

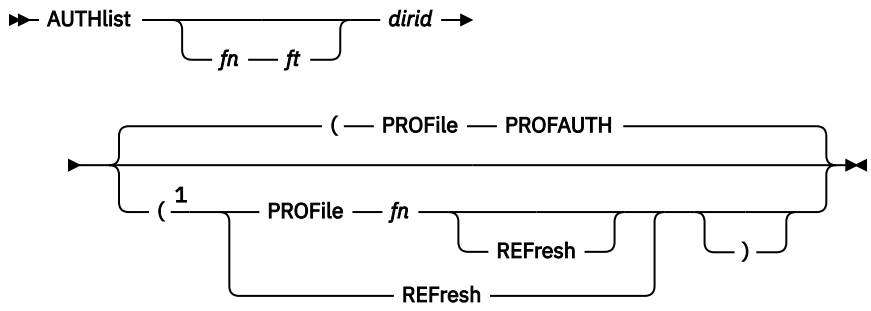
These system messages may be due to:

- Errors in command syntax

To display these messages, enter:

```
help sysmsgs tasks
```

AUTHLIST



Notes:

[1] You can enter Options in any order between the parentheses.

Authorization

General User

Purpose

Use the AUTHLIST command only in the FILELIST and DIRLIST environments. Use the AUTHLIST command to display authority information in a full screen environment. PF6 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 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 authority to the file or directory and the authority they have.

The information is placed in an XEDIT file named *userid* AUTHLIST A0.

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, *dirid* is the name of the directory for which you want authorities displayed.

Options

PROFile *fn*
specifies the name of an XEDIT macro to be executed when XEDIT is invoked by the ALIALIST command. If not specified, the default macro PROFAUTH XEDIT is invoked. For more information on the PROFAUTH macro, see Usage Note “3” on page 1390.

REFresh
updates the display with any changes to the authority information since the last update.

Usage Notes

1. For more information on how to customize this command see Appendix A, “Customizing Profiles for CMS Productivity Aids,” on page 1419.
2. 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.

3. Default PF Key Settings for AUTHLIST

   Entering the AUTHLIST command executes the PROFAUTH XEDIT macro. It sets the keys to the following values (the AUTHLIST command is only valid for files in SFS directories):

Key	Setting	Action
PF1	Help	Displays the AUTHLIST command description.
PF2	Refresh	Updates the list to display new or changed authority information.
PF3	Return	Return to the FILELIST, or DIRLIST screen.
PF4	S(Grantee)	Sort the list by grantee.
PF5	Sort (W)	Sort the list by write authority and grantee.
PF6	Sort(R)	Sort the list by read authority. (PF6 is set only when AUTHLIST is entered for a FILECONTROL directory. Otherwise, PF6 is not assigned.)
PF7	Backward	Scroll backward one screen.
PF8	Forward	Scroll forward one screen.
PF9		Unassigned.
PF10		Unassigned.
PF11	Sort(NW)	Sort the list by NEWWRITE authority. (PF11 is set only when AUTHLIST is entered for a FILECONTROL directory. Otherwise, PF11 is not assigned.)
PF12	Sort(NR)	Sort the list by NEWREAD authority. (PF12 is set only when AUTHLIST is entered for a FILECONTROL directory. Otherwise, PF12 is not assigned.)

   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. In addition to setting these PF keys, the PROFAUTH XEDIT macro sets synonyms you can use to sort your AUTHLIST screen.

   These synonyms are:

   SDIRREAD
   Sorts the list by DIRREAD authority.

   SDIRWRITE
   Sorts the list by DIRWRITE authority.

   SGRANTEE
   Sorts the list alphabetically by grantee.

   SNEWREAD
   Sorts the list by NEWREAD authority.

   SNEWWRITE
   Sorts the list by NEWWRITE authority.

   SREAD
   Sorts the list by read authority.

   SWRITE
   Sorts the list by write authority.

4. For DIRCONTROL directories, two columns having the headings DR (for DIRREAD) and DW (for DIRWRITE) are displayed. These columns are displayed only when AUTHLIST is issued for a directory

(that is, when *dirid* is specified without *fn ft*). When DR or DW contain an X, the corresponding R or W will also contain an X because these authorities are derived from the directory control authority.

5. For FILECONTROL directories, two columns having the headings NR (for NEWREAD), and NW (for NEWWRITE) are displayed. These columns are displayed only when AUTHLIST is issued for a directory (that is, when *dirid* is specified without *fn ft*). The R and W columns are independent of the NR and NW columns.
6. When *fn ft* is specified, the display shows read (R) or write (W) authority for the indicated file(s), which, in the case of files in a directory with the DIRCONTROL attribute, are derived from the DIRREAD and or DIRWRITE levels of authority.
7. 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.

Note: For more information on the SFS authorizations, see "GRANT AUTHORITY" on page 375.

Examples

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.

```
 JONES    AUTHLIST A0  V 165  Trunc=165  Size=4 Line=1 Col=1 Alt=0
File = CMSFILES SCRIPT FPOOL1:MILLER.GOODIES.FOOD
Grantee  R  W
EDWARDS  X  -
JONES    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 91. Sample AUTHLIST Screen

For more examples on using the AUTHLIST command, see *z/VM: CMS User's Guide*.

Messages and Return Codes

- DMS651E AUTHLIST must be issued from FILELIST or DIRLIST [RC=40]
- DMS653E Error executing QUERY AUTHORITY, rc=*nn* [RC=*nn*]
- DMS1132E Invalid number of operands [RC=24]
- DMS1230E AUTHLIST is invalid for minidisk [file] [RC=0]
- DMS1233E Invalid use of REFRESH option [RC=40]

Additional system messages may be issued by this command. The reasons for these messages and their location are:

Reason	Location
Errors in command syntax	"Command Syntax Error Messages" on page 1411

DISCARD

Authorization

General User

Purpose

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). Use this command only in the FILELIST, MACLIST, DIRLIST, RDRLIST, and PEEK command environments.

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* because only one reader file is displayed.

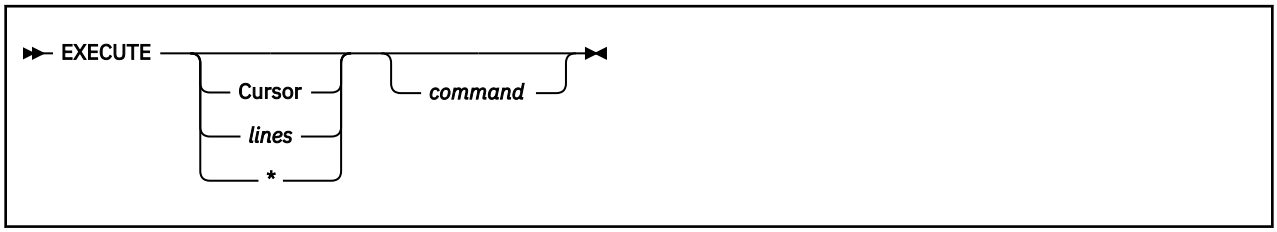
In the RDRLIST environment, *objectid* is:

spoolid
is the spoolid of the reader file to be purged.

Usage Notes

1. For more information, see “ERASE” on page 212.

EXECUTE



Authorization

General User

Purpose

Use the EXECUTE command only in the CSLLIST, CSLMAP, DIRLIST, FILELIST, MACLIST, RDRLIST, and VMLINK 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 Enter. 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.

Operands

Cursor
means 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. 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, as shown in Figure 92 on page 1394.

```
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 Enter, 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
  ..................
```

Figure 92. Sample FILELIST Screen

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
   specifies the number of lines the command is executed for, starting with the current line.

   command
   specifies 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. If no symbol is entered, the slash (/) symbol is the default and is appended to the end of the command. For more information about using symbols as part of a command, see the usage notes of each of the commands that use EXECUTE.

4. When in FILELIST, one of the most powerful features of FILELIST can be used:

   ```
   EXECUTE * COPYFILE / = = Z (OLDDATE
   ```

If you wanted to handle EXEC and XEDIT files, you could do the following:

```
FILELIST * EXEC A                    Starts FILELIST
FILELIST * XEDIT A (APPEND           Add all XEDIT files to the list,
                                      issued from FILELIST's command
                                      line
EXECUTE * COPYFILE / = = Z (OLDDATE  Also executed from the command
                                      line
```

You can also exploit the fact that FILELIST is XEDIT based, so before the EXECUTE *, you could change the list of files (for example, remove all files starting with PROF). All commands below are entered in FILELIST's command line.

```
ALL / PROF/       Show all files with filenames beginning with: PROF
DEL *             Remove them from the FILELIST list.
                   (This does not erase the files on the disk)
ALL               Display all lines again
TOP               Go to the top
EXECUTE * ...
```

Messages and Return Codes

- DMS526E Option CURSOR valid in display mode only [RC=3]
- DMS543E Invalid number: *number* [RC=5]
- DMS561E Cursor is not on a valid data field [RC=3]
- DMS651E EXECUTE must be issued from CSLLIST, CSLMAP, DIRLIST, FILELIST, MACLIST, RDRLIST, or VMLINK [RC=40]
- DMS654E 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 75 on page 1397 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 file type changed from ‘HELPxxxx’ to ‘$HLPxxxx’.

Table 75. HELP and HELPCONV Format Word Summary

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 file name. 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

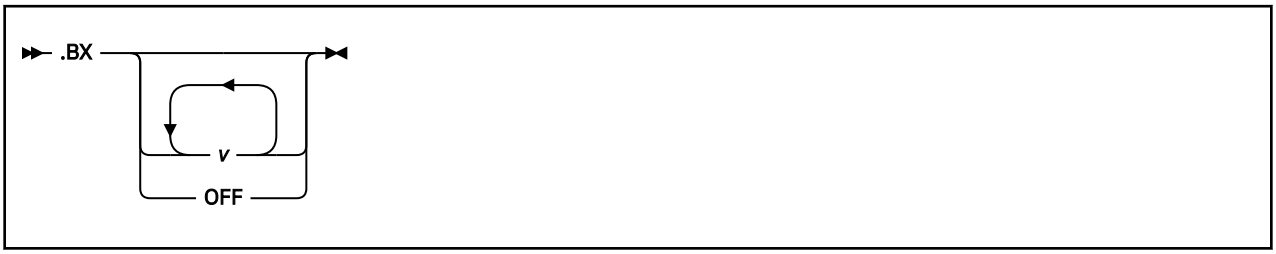
Table 75. HELP and HELPCONV Format Word Summary (continued)

FORMAT WORD	OPERAND FORMAT	FUNCTION	BREAK	DEFAULT VALUE
.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.

Format



Operands

v
is a position 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.

Usage Notes

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 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.

Format

Operand

comment
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 *comment* 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

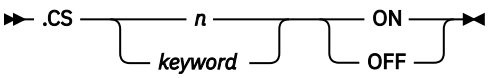
```
.CM Remember to change the date.
```

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 to be displayed based on the specified HELP command option.

Format



Operands

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:

1. BRIEF
2. DESCRIPT
3. FORMAT
4. PARMS
5. OPTIONS
6. NOTES
7. ERRORS
8. 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 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:

```
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.

Note:

1. Few RELATED HELP files exist, but you can use this option to customize your own HELP files.
2. 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.

Format

Operands

ON
restores default HELPCONV formatting, including both justification and concatenation of lines. The default is ON.

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 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.

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

   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.

Format

Operand

n

specifies the number of character spaces to shift the next line from the current margin. An operand of +n specifies 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), error message DMS5251E 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.

Format

Operand

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.

Format

Operand

component
is the component used by the menu.

Usage Notes

1. The .MT format word assists 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 file type "HELPFUNC". The .MT control word defines a component ID to override that derived from the file name. The FUNCTION menu could be:

```
.MT REXX
```

to specify the menu contains a list of REXX commands and thus will be found under the file type “HELPREXX”.

.OF (OFFSET)

Use the OFFSET format word to indent all but the first line of a block of text.

Format

Operand

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: The .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:

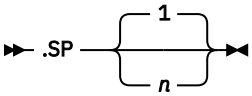
```
.of
```

   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.

Format



Operand

n
specifies the number of blank lines to be inserted in the output. The default is 1, if omitted.

.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 all exclamation points in the file appear as blanks in the output.

Format

Operands

source
is a source character. It may be a single character or a two-character hexadecimal code.

result
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. Because 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 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, such as those listed below, if you have specified a command format incorrectly. These error messages are prefixed by DMSPCL, and have a return code of 24.

To resolve the error, check the format of the command, make the necessary correction, and enter the command again.

For example, if you issue the CREATE FILE command as CREATE, you receive the message:

```
DMSPCL384E  Missing modifier keyword(s)
```

Check the format of the CREATE FILE command and enter it again, using the correct command name and operands.

Table 76. Command Syntax Error Messages

Number	Message
384E	Missing modifier keyword(s)
385E	Invalid modifier keyword: *keyword*
386E	Missing operand(s)
387E	Missing value for *operand* operand
387E	Missing alphanumeric string for *operand* operand
387E	Missing application identifier for *operand* operand
387E	Missing character for *operand* operand
387E	Missing device address for *operand* operand
387E	Missing filename for *operand* operand
387E	Missing filetype for *operand* operand
387E	Missing execname for *operand* operand
387E	Missing exectype for *operand* operand
387E	Missing filemode for *operand* operand
387E	Missing hexadecimal number for *operand* operand
387E	Missing integer for *operand* operand
387E	Missing negative integer for *operand* operand
387E	Missing number for *operand* operand
387E	Missing positive integer for *operand* operand

Table 76. Command Syntax Error Messages (continued)

Number	Message
387E	Missing mode for *operand* operand
387E	Missing character string for *operand* operand
387E	Missing directory id for *operand*
387E	Missing namedef for *operand*
387E	Missing filepoolid for *operand*
387E	Missing filemode, filepoolid or directory id for *operand* operand
388E	Invalid keyword: *keyword*
389E	Invalid operand: *operand*
389E	Invalid alphanumeric string: *string*
389E	Invalid application identifier: *string*
389E	Invalid character: *character*
389E	Invalid device address: *address*
389E	Invalid fileid: *filename filetype*
389E	Invalid filename: *filename*
389E	Invalid filetype: *filetype*
389E	Invalid execname: *execname*
389E	Invalid exectype: *exectype*
389E	Invalid filemode: *filemode*
389E	Invalid hexadecimal number: *number*
389E	Invalid integer: *integer*
389E	Invalid negative integer: *integer*
389E	Invalid number: *integer*
389E	Invalid positive integer: *integer*
389E	Invalid mode: *mode*
389E	Invalid character string: *string*
389E	Invalid directory id: *dirid*
389E	Invalid namedef: *namedef*
389E	Invalid filepoolid: *filepoolid*
389E	Invalid filemode or directory id: *value*
389E	Missing filemode, filepoolid or directory id: *value*
389E	Invalid fileid: *filename filetype*
390E	Invalid value *value* for *operand* operand
390E	Invalid alphanumeric string *string* for *operand* operand
390E	Invalid application identifier *applid* for *operand* operand
390E	Invalid character *character* for *operand* operand
390E	Invalid device address *address* for *operand* operand

Table 76. Command Syntax Error Messages (continued)

Number	Message
390E	Invalid filename *filename* for *operand* operand
390E	Invalid filetype *filetype* for *operand* operand
390E	Invalid execname *execname* for *operand* operand
390E	Invalid exectype *exectype* for *operand* operand
390E	Invalid filemode *filemode* for *operand* operand
390E	Invalid hexadecimal number *number* for *operand* operand
390E	Invalid integer *integer* for *operand* operand
390E	Invalid negative integer *integer* for *operand* operand
390E	Invalid number *number* for *operand* operand
390E	Invalid positive integer *integer* for *operand* operand
390E	Invalid mode *mode* for *operand* operand
390E	Invalid character string *string* for *operand* operand
390E	Invalid directory id *dirid* for *operand* operand
390E	Invalid namedef *namedef* for *operand* operand
390E	Invalid filepoolid *filepoolid* for *operand* operand
390E	Invalid filemode or directory id *value* for *operand* operand
390E	Missing filemode, filepoolid or directory id *value* for operand *operand*
391E	Unexpected operand(s): *operand*
393E	Missing value for *option* option
393E	Missing alphanumeric string for *option* option
393E	Missing application identifier for *option* option
393E	Missing character for *option* option
393E	Missing device address for *option* option
393E	Missing filename for *option* option
393E	Missing filetype for *option* option
393E	Missing execname for *option* option
393E	Missing exectype for *option* option
393E	Missing filemode for *option* option
393E	Missing hexadecimal number for *option* option
393E	Missing integer for *option* option
393E	Missing negative integer for *option* option
393E	Missing number integer for *option* option
393E	Missing positive integer for *option* option
393E	Missing mode for *option* option
393E	Missing character string for *option* option
393E	Missing directory id for *option* option

Table 76. Command Syntax Error Messages (continued)

Number	Message
393E	Missing namedef for *option* option
393E	Missing filepoolid for *option* option
393E	Missing filemode or directory id for *option* option
393E	Missing filemode, filepoolid or directory id for *option* option
394E	Invalid option: *option*
395E	Invalid value *value* for *option* option
395E	Invalid alphanumeric string *string* for *option* option
395E	Invalid application identifier *applid* for *option* option
395E	Invalid character *character* for *option* option
395E	Invalid device address *address* for *option* option
395E	Invalid filename *filename* for *option* option
395E	Invalid filetype *filetype* for *option* option
395E	Invalid execname *execname* for *option* option
395E	Invalid exectype *exectype* for *option* option
395E	Invalid filemode *filemode* for *option* option
395E	Invalid hexadecimal number *number* for *option* option
395E	Invalid integer *integer* for *option* option
395E	Invalid number *number* for *option* option
395E	Invalid negative integer *integer* for *option* option
395E	Invalid positive integer *integer* for *option* option
395E	Invalid mode *mode* for *option* option
395E	Invalid character string *string* for *option* option
395E	Invalid directory id *dirid* for *option* option
395E	Invalid namedef *namedef* for *option* option
395E	Invalid filepoolid *filepoolid* for *option* option
395E	Invalid filemode or directory id *value* for *option* option
395E	Missing filemode, filepoolid or directory id *value* for *option* option

File Pool Server Messages

If you enter a command that interacts with a file pool server you may receive error messages such as those listed below. For more information on messages and the suggested action to resolve the error, see *z/VM: CMS and REXX/VM Messages and Codes*.

Table 77. File Pool Server Messages

Number	Message
109S	Virtual storage capacity exceeded [RC=31|104]
1137E	Object is locked or in use, or there is an outstanding lock or disable in the object's directory hierarchy [RC=31|70]

Table 77. File Pool Server Messages (continued)

Number	Message
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 exceeded for file pool *filepoolid* [RC=0]
1141W	User filespace threshold exceeded [RC=0]
1142E	Error reading system catalog for file pool *filepoolid*; error codes *nn* and *nn*; Detecting module *moduleid* [RC=31|104]
1142E	Error writing system catalog for file pool *filepoolid*; error codes *nn* and *nn*; Detecting module *moduleid* [RC=31|104]
1142E	Error in file access function for file pool *filepoolid*; error codes *nn* and *nn*; Detecting module *moduleid* [RC=31|104]
1142E	Error in locking function for file pool *filepoolid*; error codes *nn* and *nn*; Detecting module *moduleid* [RC=31|104]
1142E	Error in query function for file pool *filepoolid*; error codes *nn* and *nn*; Detecting module *moduleid* [RC=31|104]
1142E	Error in storage management for file pool *filepoolid*; error codes *nn* and *nn*; Detecting module *moduleid* [RC=31|104]
1142E	Error *nn* for file pool *filepoolid*; error codes *nn* and *nn*; Detecting module *moduleid* [RC=31|104]
1142E	Error *nn* in SFS adapter routine; error codes *nn* and *nn*; Detecting module *moduleid* [RC=31|104]
1143E	Inconsistent catalogs in file pool *filepoolid*; error codes *nn* and *nn*; Detecting module *moduleid* [RC=31|104]
1144E	Implicit rollback occurred for work unit *workunitid* [RC=31|104]
1145E	Further communication with file pools is impossible [RC=31|104]
1146E	Deadlock *code* encountered for file pool *filepoolid* [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 accounting exit routine [RC=31|40]
1151E	File pool *filepoolid* is unavailable [RC=31|55]
1153E	File pool *filepoolid* is unavailable or unknown [RC=31|99]
1155E	CSL routine *cslname* is not loaded [RC=40]
1155E	CSL routine *cslname* has been dropped [RC=40]
1155E	Secondary CSL routine *cslname* is not loaded, or has been dropped [RC=40]
1156S	Supervisor error 1: return code *nn* [RC=31|104]
1156S	Supervisor error 2: return code *nn*, reason code *nn* [RC=31|104]

Table 77. File Pool Server Messages (continued)

Number	Message
1157E	Work unit already active when atomic request is issued for work unit *workunitid* [RC=31 | 70]
1158E	Attempt to make uncommitted updates to more than one file pool on work unit *workunitid* [RC=31 | 70]
1159E	User has files or directories open when a COMMIT is requested on work unit *workunitid* [RC=31 | 70]
1164E	Request *cslname* failed; storage group being restored [RC=31 | 99]
1170W	The maximum number of APPC/VM connections allowed for your userid was exceeded. An inactive communication path was severed in order to establish a new path for your request. This will recur with each additional request beyond your allowed maximum. [RC=4]
1174E	The MAXCONN limit has been reached. You have tried to establish more APPC/VM connections than is allowed for your userid. There are no inactive communication paths available for reuse for the current request [RC=55 | 31]
1174E	Your attempt exceeds the number of APPC/VM connections allowed for file pool *filepoolid* [RC=31 | 55]
1176E	Virtual storage capacity exceeded for file pool *filepoolid* [RC=31 | 99]
1179E	*filepoolid* is a remote file pool that was started for local use only [RC=31 | 99]
1200E	Operation failed due to code-level mismatch of CMS and file pool *filepoolid* [RC=31 | 88]
1212E	You have opened a file pool catalog for WRITE on work unit *workunitid* for file pool *filepoolid* [RC=31 | 40]
1217E	Rollback occurred during CMS command processing [RC=31]
1217E	Rollback occurred during CMS end-of-command processing [RC=31]
1223E	There is no default file pool currently defined [RC=40]
1240E	You are not authorized to connect to file pool *filepoolid* [RC=31 | 76]
1254E	An attempt to commit will exceed the number of 4K blocks allowed for the user in file pool *filepoolid* [RC=31 | 40]
1259E	File pool *filepoolid* has run out of physical space in the storage group [RC=31]
1265E	A commit or rollback was in process when a request was issued for work unit *workunitid* [RC=31 | 70]
1311E	Object already exists [RC=28 | 31]
2001E	CRR resynchronization processing is attempting to complete a commit of the changes [RC=4 | 31]
2001E	CRR resynchronization processing is attempting to complete a rollback of the changes [RC=4 | 31]
2002W	The processing environment has changed [RC=4]
2003E	Changes were rolled back. The CRR recovery server is not available [RC=31]
2004E	There is a data integrity problem. Some changes were committed and some changes were rolled back [RC=104]
2005E	There may be a data integrity problem. Some of the changes were committed, however, some changes may have been rolled back [RC=104]

Table 77. File Pool Server Messages (continued)

Number	Message
2005E	There may be a data integrity problem. Some changes may have been committed and some changes were rolled back [RC=104]
2006E	One or more non-recoverable files could not be closed during rollback processing [RC=31]
2007E	One or more resources are not in a correct state to commit or roll back changes [RC=40]
2008E	Error establishing communications between CRR recovery server and file pool *filepoolid*. Error codes *nn* and *nn*. Detecting module *moduleid*
2009E	CRR recovery server log record exceeded the maximum allowed protected resources [RC=31]
2010E	An attempt to write to file pool *filepoolid* was rejected. Only one write mode resource is allowed for the work unit; one already is in write mode [RC=40]
2013E	File system capacity exceeded; number of physical blocks in file exceeds system limit [RC=31|88]
2016E	No CRR commits will be allowed until CRR recovery server log space is available [RC=31|99]
2023E	File pool *filepoolid* does not support the requested command. [RC=88]
2026E	File sharing conflict with resynchronization activity in file pool *filepoolid*. Recovery token *token* [RC=31|70]
2031E	Mixing operations on SFS objects and BFS objects is not valid within a single work unit [RC={31|70}]
2031E	Mixing recoverable and non-recoverable work to a file is not valid within a single work unit. [RC={31|70}]
2148W	One or more unresolved aliases can not be resolved
2149E	File *fileid* is migrated and DFSMS/VM recall processing is not active [RC = 51]
2153E	File *fileid* is migrated and DFSMS/VM is not available [RC = 51]
2154E	File *fileid* is migrated and implicit RECALL is set to OFF [RC = 40|50]
2155E	SFS error *reascode* in file pool *filepoolid* occurred during recall of file *fileid* [RC = 51]
2510E	Requested function is not supported for specified file object [RC=40|31]
2523E	Unexpected SFS reason code *nn*; return code *nn*; secondary error codes *nn* and *nn*. Detecting module *moduleid* [RC=104]
2523E	Unexpected SFS reason code *nn*; return code *nn*; secondary error codes *nn* and *nn*. Detecting module *moduleid*. The server for file pool *filepoolid* is at a higher service level than CMS in your virtual machine. [RC=104]
2523E	Unexpected SFS reason code *nn*; return code *nn*; secondary error codes *nn* and *nn*. The server for file pool *filepoolid* is at a higher service level than CMS in your virtual machine. [RC=104]
2524E	Concurrent use of multiple file pool identifiers that resolve to file pool *filepoolid* [RC=40]
2525E	System error in DFSMS/VM [RC=51]
2526E	File or directory creation or file recall was rejected by a DFSMS/VM ACS routine; ACS routine return code *nnnnn* [RC=51]
2528E	Communication error in DFSMS/VM [RC=51]

Table 77. File Pool Server Messages (continued)

Number	Message
2530E	No file blocks are assigned for this user in file pool *filepoolid* [RC=8]
2536E	File space usage exit caused a rollback to occur for file pool *filepoolid* [RC=31]

File Error Messages

If you enter a command that uses or changes files, you may receive one of the error messages listed below. For more information on the message and the suggested action to resolve the error, see *z/VM: CMS and REXX/VM Messages and Codes*.

Table 78. File Error Messages

Number	Message
005E	No filename|filetype specified [RC=20]
002E	File *fn ft fm* not found [RC=28]
024E	File *fn ft fm* already exists [RC=35]
030E	File *fn ft fm* already active [RC=37]
037E	Filemode *fm* is accessed as read/only [RC=12]
048E	Invalid filemode *mode* [RC=24]
062E	Invalid character *char* in fileid *fn ft [fm]* [RC=20]
062E	SO and SI are invalid fileid characters [RC=20]
069E	Filemode *mode* not accessed [RC=36]
107S	Disk *mode(vdev)* is full
107S	File space *filespacename* in file pool *filepoolid* is full
109S	Insufficient free storage available [RC=25]
1138E	File sharing conflict involving file *fn*[RC=70]
1180E	You have an explicit lock on file *fn ft fm*, the erase failed [RC=70]
1184E	File *fn ft fm* not found or not authorized [RC=28]
1258E	You are not authorized to write to file *fn ft fm* [RC=28]
1260E	Invalid OPENTYP *xx* specified in FSCB for file *fn ft fm* [RC=33]
1261E	Invalid CACHE specified in FSCB for file *fn ft fm* [RC=34]
1262S	Error *nn* opening file *fn ft fm* [RC=3 | 11 | 80 | 81 | 82 | 83 | 84]
3362E	Imbedded blanks found in filename|filetype [RC=20]

Appendix A. Customizing Profiles for CMS Productivity Aids

This appendix shows an example of how to customize the PROFILEs used for the CMS productivity aids. When making updates to your PROFILE, IBM recommends you do not use the following command defaults:

```
Alialist        options = PROFILE PROFALIA
Authlist        options = PROFILE PROFAUTH
Csllist         options = PROFILE PROFCLST
Cslmap          options = PROFILE PROFCSLM
Dirlist         options = PROFILE PROFDLST
Filelist        options = PROFILE PROFFLST PROFILE2 PROFFSHR
                          PROFILE3 PROFFSEA PROFILE4 PROFFDAT
Maclist         options = PROFILE PROFMLST
Note            options = PROFILE PROFNOTE
Peek            options = PROFILE PROFPEEK
Rdrlist         options = PROFILE PROFRLST
Vmlink          options = PROFILE PROFVMLK
```

instead you should write your own profile which calls the IBM-supplied profile and performs the customization.

For example, if you want to change your RDRLIST screen, you need to create your own customized profile as shown in the "Customized RDRLIST Profile Example" below. You will use the defaults command, to run your profile, and your profile will call the IBM-supplied profile, PROFRLST. After PROFRLST returns, the user profile then continues with the customized changes. Creating a user profile in this manner will not eliminate the problems encountered when new function is added however, doing this will better ensure upward compatibility between releases.

To use your customized RDRLIST profile, Enter:

```
DEFAULTS SET RDRLIST PROFILE PROFRUSR
```

Note: Because certain productivity aids stack some XEDIT subcommands, consider stacking your customized profile as:

```
/* MYPEEK XEDIT /*
queue 'macro MYPEEK2'
exit
```

This will eliminate the possibility of having your settings overwritten by the productivity aids. For more information, see the specific productivity aid.

Customized RDRLIST Profile Example

```
/* User Customized profile for RDRLIST */
  parse arg rdrlist_parameters         /* get parms */
  'MACRO PROFRLST' rdrlist_parameters  /* call IBM-supplied with parms */

/* Define PF key 10 to be Sort by name */
  'EXTRACT /RESERVED */'               /* get all reserved lines */
/* find the correct line to change by looking for "10=" */
  do i = 1 to reserved.0
     /* get each reserved line */
     parse var reserved.i line color ext_high pss high text
     pf10_pos = 'POS'('10=',text)      /* find position of change */
     if pf10_pos = 0 then iterate
     /* change the line */
     newtext = 'OVERLAY'('Sort(name)',text,pf10_pos+4,10)
     /* put new line on screen */
     'SET RESERVED' line color ext_high pss high newtext
     leave /* found desired line; leave loop */
     end
'SET PF10 SNAME#COMMAND CURSOR COLUMN'  /* set PF10 function */

/* Set the file identification line color to BLUE with reverse video */
```

```
   'SET COLOR IDLINE BLUE REVVIDEO'

/* Set the command line to be turquoise */
   'SET COLOR CMDLINE TURQ'

   exit
```

Notices

This information was developed for products and services offered in the US. This material might be available from IBM in other languages. However, you may be required to own a copy of the product or product version in that language in order to access it.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
US

For license inquiries regarding double-byte character set (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some jurisdictions do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you provide in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
US

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

The performance data and client examples cited are presented for illustrative purposes only. Actual performance results may vary depending on specific configurations and operating conditions.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information may contain examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to actual people or business enterprises is entirely coincidental.

COPYRIGHT LICENSE:

This information may contain sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Programming Interface Information

This book documents information NOT intended to be used as Programming Interfaces of z/VM.

This book also documents intended Programming Interfaces that allow the customer to write programs to obtain the services of z/VM. This information is identified where it occurs, either by an introductory statement to a chapter or section or by the following marking:

PI

<...Programming Interface information...>

PI end

Trademarks

IBM, the IBM logo, and ibm.com® are trademarks or registered trademarks of International Business Machines Corp., in the United States and/or other countries. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on IBM Copyright and trademark information (https://www.ibm.com/legal/copytrade).

The registered trademark Linux is used pursuant to a sublicense from the Linux Foundation, the exclusive licensee of Linus Torvalds, owner of the mark on a worldwide basis.

Terms and Conditions for Product Documentation

Permissions for the use of these publications are granted subject to the following terms and conditions.

Applicability

These terms and conditions are in addition to any terms of use for the IBM website.

Personal Use

You may reproduce these publications for your personal, noncommercial use provided that all proprietary notices are preserved. You may not distribute, display or make derivative work of these publications, or any portion thereof, without the express consent of IBM.

Commercial Use

You may reproduce, distribute and display these publications solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of these publications, or reproduce, distribute or display these publications or any portion thereof outside your enterprise, without the express consent of IBM.

Rights

Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the publications or any information, data, software or other intellectual property contained therein.

IBM reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the publications is detrimental to its interest or, as determined by IBM, the above instructions are not being properly followed.

You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations.

IBM MAKES NO GUARANTEE ABOUT THE CONTENT OF THESE PUBLICATIONS. THE PUBLICATIONS ARE PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, AND FITNESS FOR A PARTICULAR PURPOSE.

IBM Online Privacy Statement

IBM Software products, including software as a service solutions, ("Software Offerings") may use cookies or other technologies to collect product usage information, to help improve the end user experience, to tailor interactions with the end user, or for other purposes. In many cases no personally identifiable information is collected by the Software Offerings. Some of our Software Offerings can help enable you to collect personally identifiable information. If this Software Offering uses cookies to collect personally identifiable information, specific information about this offering's use of cookies is set forth below.

This Software Offering does not use cookies or other technologies to collect personally identifiable information.

If the configurations deployed for this Software Offering provide you as customer the ability to collect personally identifiable information from end users via cookies and other technologies, you should seek your own legal advice about any laws applicable to such data collection, including any requirements for notice and consent.

For more information about the use of various technologies, including cookies, for these purposes, see:

- The section entitled **IBM Websites** at IBM Privacy Statement (https://www.ibm.com/privacy)
- Cookies and Similar Technologies (https://www.ibm.com/privacy#Cookies_and_Similar_Technologies)

Bibliography

This topic lists the publications in the z/VM library. For abstracts of the z/VM publications, see *z/VM: General Information*.

Where to Get z/VM Information

The current z/VM product documentation is available in IBM Documentation - z/VM (https://www.ibm.com/docs/en/zvm).

z/VM Base Library

Overview

- *z/VM: License Information*, GI13-4377
- *z/VM: General Information*, GC24-6286

Installation, Migration, and Service

- *z/VM: Installation Guide*, GC24-6292
- *z/VM: Migration Guide*, GC24-6294
- *z/VM: Service Guide*, GC24-6325
- *z/VM: VMSES/E Introduction and Reference*, GC24-6336

Planning and Administration

- *z/VM: CMS File Pool Planning, Administration, and Operation*, SC24-6261
- *z/VM: CMS Planning and Administration*, SC24-6264
- *z/VM: Connectivity*, SC24-6267
- *z/VM: CP Planning and Administration*, SC24-6271
- *z/VM: Getting Started with Linux on IBM Z*, SC24-6287
- *z/VM: Group Control System*, SC24-6289
- *z/VM: I/O Configuration*, SC24-6291
- *z/VM: Running Guest Operating Systems*, SC24-6321
- *z/VM: Saved Segments Planning and Administration*, SC24-6322
- *z/VM: Secure Configuration Guide*, SC24-6323

Customization and Tuning

- *z/VM: CP Exit Customization*, SC24-6269
- *z/VM: Performance*, SC24-6301

Operation and Use

- *z/VM: CMS Commands and Utilities Reference*, SC24-6260
- *z/VM: CMS Primer*, SC24-6265
- *z/VM: CMS User's Guide*, SC24-6266
- *z/VM: CP Commands and Utilities Reference*, SC24-6268

- *z/VM: System Operation*, SC24-6326
- *z/VM: Virtual Machine Operation*, SC24-6334
- *z/VM: XEDIT Commands and Macros Reference*, SC24-6337
- *z/VM: XEDIT User's Guide*, SC24-6338

Application Programming

- *z/VM: CMS Application Development Guide*, SC24-6256
- *z/VM: CMS Application Development Guide for Assembler*, SC24-6257
- *z/VM: CMS Application Multitasking*, SC24-6258
- *z/VM: CMS Callable Services Reference*, SC24-6259
- *z/VM: CMS Macros and Functions Reference*, SC24-6262
- *z/VM: CMS Pipelines User's Guide and Reference*, SC24-6252
- *z/VM: CP Programming Services*, SC24-6272
- *z/VM: CPI Communications User's Guide*, SC24-6273
- *z/VM: ESA/XC Principles of Operation*, SC24-6285
- *z/VM: Language Environment User's Guide*, SC24-6293
- *z/VM: OpenExtensions Advanced Application Programming Tools*, SC24-6295
- *z/VM: OpenExtensions Callable Services Reference*, SC24-6296
- *z/VM: OpenExtensions Commands Reference*, SC24-6297
- *z/VM: OpenExtensions POSIX Conformance Document*, GC24-6298
- *z/VM: OpenExtensions User's Guide*, SC24-6299
- *z/VM: Program Management Binder for CMS*, SC24-6304
- *z/VM: Reusable Server Kernel Programmer's Guide and Reference*, SC24-6313
- *z/VM: REXX/VM Reference*, SC24-6314
- *z/VM: REXX/VM User's Guide*, SC24-6315
- *z/VM: Systems Management Application Programming*, SC24-6327
- *z/VM: z/Architecture Extended Configuration (z/XC) Principles of Operation*, SC27-4940

Diagnosis

- *z/VM: CMS and REXX/VM Messages and Codes*, GC24-6255
- *z/VM: CP Messages and Codes*, GC24-6270
- *z/VM: Diagnosis Guide*, GC24-6280
- *z/VM: Dump Viewing Facility*, GC24-6284
- *z/VM: Other Components Messages and Codes*, GC24-6300
- *z/VM: VM Dump Tool*, GC24-6335

z/VM Facilities and Features

Data Facility Storage Management Subsystem for z/VM

- *z/VM: DFSMS/VM Customization*, SC24-6274
- *z/VM: DFSMS/VM Diagnosis Guide*, GC24-6275
- *z/VM: DFSMS/VM Messages and Codes*, GC24-6276
- *z/VM: DFSMS/VM Planning Guide*, SC24-6277

- *z/VM: DFSMS/VM Removable Media Services*, SC24-6278
- *z/VM: DFSMS/VM Storage Administration*, SC24-6279

Directory Maintenance Facility for z/VM

- *z/VM: Directory Maintenance Facility Commands Reference*, SC24-6281
- *z/VM: Directory Maintenance Facility Messages*, GC24-6282
- *z/VM: Directory Maintenance Facility Tailoring and Administration Guide*, SC24-6283

Open Systems Adapter

- Open Systems Adapter/Support Facility on the Hardware Management Console (https://www.ibm.com/docs/en/SSLTBW_2.3.0/pdf/SC14-7580-02.pdf), SC14-7580
- Open Systems Adapter-Express ICC 3215 Support (https://www.ibm.com/docs/en/zos/2.3.0?topic=osa-icc-3215-support), SA23-2247
- Open Systems Adapter Integrated Console Controller User's Guide (https://www.ibm.com/docs/en/SSLTBW_2.3.0/pdf/SC27-9003-02.pdf), SC27-9003
- Open Systems Adapter-Express Customer's Guide and Reference (https://www.ibm.com/docs/en/SSLTBW_2.3.0/pdf/ioa2z1f0.pdf), SA22-7935

Performance Toolkit for z/VM

- *z/VM: Performance Toolkit Guide*, SC24-6302
- *z/VM: Performance Toolkit Reference*, SC24-6303

RACF Security Server for z/VM

- *z/VM: RACF Security Server Auditor's Guide*, SC24-6305
- *z/VM: RACF Security Server Command Language Reference*, SC24-6306
- *z/VM: RACF Security Server Diagnosis Guide*, GC24-6307
- *z/VM: RACF Security Server General User's Guide*, SC24-6308
- *z/VM: RACF Security Server Macros and Interfaces*, SC24-6309
- *z/VM: RACF Security Server Messages and Codes*, GC24-6310
- *z/VM: RACF Security Server Security Administrator's Guide*, SC24-6311
- *z/VM: RACF Security Server System Programmer's Guide*, SC24-6312
- *z/VM: Security Server RACROUTE Macro Reference*, SC24-6324

Remote Spooling Communications Subsystem Networking for z/VM

- *z/VM: RSCS Networking Diagnosis*, GC24-6316
- *z/VM: RSCS Networking Exit Customization*, SC24-6317
- *z/VM: RSCS Networking Messages and Codes*, GC24-6318
- *z/VM: RSCS Networking Operation and Use*, SC24-6319
- *z/VM: RSCS Networking Planning and Configuration*, SC24-6320

TCP/IP for z/VM

- *z/VM: TCP/IP Diagnosis Guide*, GC24-6328
- *z/VM: TCP/IP LDAP Administration Guide*, SC24-6329
- *z/VM: TCP/IP Messages and Codes*, GC24-6330

- *z/VM: TCP/IP Planning and Customization*, SC24-6331
- *z/VM: TCP/IP Programmer's Reference*, SC24-6332
- *z/VM: TCP/IP User's Guide*, SC24-6333

Prerequisite Products

Device Support Facilities

- Device Support Facilities (ICKDSF): User's Guide and Reference (https://www.ibm.com/docs/en/SSLTBW_2.5.0/pdf/ickug00_v2r5.pdf), GC35-0033

Environmental Record Editing and Printing Program

- Environmental Record Editing and Printing Program (EREP): Reference (https://www.ibm.com/docs/en/SSLTBW_2.5.0/pdf/ifc2000_v2r5.pdf), GC35-0152
- Environmental Record Editing and Printing Program (EREP): User's Guide (https://www.ibm.com/docs/en/SSLTBW_2.5.0/pdf/ifc1000_v2r5.pdf), GC35-0151

Related Products

XL C++ for z/VM

- *XL C/C++ for z/VM: Runtime Library Reference*, SC09-7624
- *XL C/C++ for z/VM: User's Guide*, SC09-7625

z/OS

IBM Documentation - z/OS (https://www.ibm.com/docs/en/zos)

Index

Special Characters

Numerics

A

B

C

D

E

F

G

H

I

K

L

M

N

O

P

Q

R

S

T

U

V

W

X

Y

Z

Product Number: 5741-A09

Printed in USA

SC24-6260-73