VMS DCL Dictionary

Order Number: AA–LA12A–TE

April 1988

This manual provides detailed reference information and examples on all VMS DCL commands and lexical functions.

Revision/Update Information:	This manual supersedes the VAX/VMS DCL Dictionary Version 4.4.
Software Version:	VMS Version 5.0

digital equipment corporation
maynard, massachusetts

April 1988

The postpaid READER'S COMMENTS form on the last page of this document requests the user's critical evaluation to assist in preparing future documentation.

The following are trademarks of Digital Equipment Corporation:

DEC	DIBOL	UNIBUS
DEC/CMS	EduSystem	VAX
DEC/MMS	IAS	VAXcluster
DECnet	MASSBUS	VMS
DECsystem–10	PDP	VT
DECSYSTEM–20	PDT	
DECUS	RSTS	
DECwriter	RSX	digital™

ZK9996

HOW TO ORDER ADDITIONAL DOCUMENTATION
DIRECT MAIL ORDERS

USA & PUERTO RICO*

Digital Equipment Corporation
P.O. Box CS2008
Nashua, New Hampshire
03061

CANADA

Digital Equipment
of Canada Ltd.
100 Herzberg Road
Kanata, Ontario K2K 2A6
Attn: Direct Order Desk

INTERNATIONAL

Digital Equipment Corporation
PSG Business Manager
c/o Digital's local subsidiary
or approved distributor

In Continental USA and Puerto Rico call 800-258-1710.
In New Hampshire, Alaska, and Hawaii call 603-884-6660.
In Canada call 800-267-6215.
*Any prepaid order from Puerto Rico must be placed with the local Digital subsidiary (809-754-7575).
Internal orders should be placed through the Software Distribution Center (SDC), Digital Equipment Corporation, Westminster, Massachusetts 01473.

Production Note

This book was produced with the VAX DOCUMENT electronic publishing system, a software tool developed and sold by DIGITAL. In this system, writers use an ASCII text editor to create source files containing text and English-like code; this code labels the structural elements of the document, such as chapters, paragraphs, and tables. The VAX DOCUMENT software, which runs on the VMS operating system, interprets the code to format the text, generate a table of contents and index, and paginate the entire document. Writers can print the document on the terminal or line printer, or they can use DIGITAL-supported devices, such as the LN03 laser printer and PostScript™ printers (PrintServer 40 or LN03R ScriptPrinter), to produce a typeset-quality copy containing integrated graphics.

™ PostScript is a trademark of Adobe Systems, Inc.

Contents

INDEX

TABLES

Preface

Intended Audience

This manual is intended for all users of the VMS operating system. It includes complete descriptions of all DCL commands and lexical functions. If a command has any restrictions or requires special privileges, they are noted in reference information for that command.

Document Structure

This manual contains detailed descriptions of each command and lexical function. The commands are listed in alphabetical order, with the command name appearing at the top of every page. The lexical functions are grouped under "Lexical Functions" (after the JOB command description) and are listed alphabetically within that grouping; the lexical function name appears at the top of each page.

Readers of this manual should be familiar with the material covered in the *VMS DCL Concepts Manual*. Furthermore, while the *Guide to Using VMS Command Procedures* is not a requirement for using this manual, it does help clarify some of the examples involving command procedures.

The commands that invoke language compilers and other VAX optional software products are not included in this manual; they are included in the documentation provided with those products.

Users familiar with previous releases of the *VMS DCL Dictionary* should note that the general overview of DCL command language concepts (formerly, Part 1) has been removed. For a general discussion of the DCL command language, see the *VMS DCL Concepts Manual*.

Associated Documents

For an introduction to the VMS operating system and the use of the Digital Command Language, see the *Introduction to VMS*. This manual is especially recommended for novice users or users lacking experience with interactive computer systems.

The *VMS DCL Concepts Manual* provides a general overview of DCL command language concepts.

The *Guide to Using VMS Command Procedures* defines and illustrates good practices in constructing command procedures with DCL commands and lexical functions.

The various VMS Utilities reference manuals document major VMS Utilities. These manuals describe the DCL commands that invoke the various utilities, any commands that you can enter while running a utility, and other information. For all utilities documented in these volumes, the *VMS DCL Dictionary* provides only a brief description and format information.

The *VMS System Messages and Recovery Procedures Reference Volume* explains any error and warning messages you may receive. In most cases, however, you will not need to refer to this manual, because it will be obvious from the message text and other information on your screen what action (if any) to take.

The *Overview of VMS Documentation* describes the new organization of the VMS document set. This manual shows how the individual manuals fit together and relate to each other.

Conventions

Convention	Meaning
RET	In examples, a key name (usually abbreviated) shown within a box indicates that you press a key on the keyboard; in text, a key name is not enclosed in a box. In this example, the key is the RETURN key. (Note that the RETURN key is not usually shown in syntax statements or in all examples; however, assume that you must press the RETURN key after entering a command or responding to a prompt.)
CTRL/C	A key combination, shown in uppercase with a slash separating two key names, indicates that you hold down the first key while you press the second key. For example, the key combination CTRL/C indicates that you hold down the key labeled CTRL while you press the key labeled C. In examples, a key combination is enclosed in a box.
$ SHOW TIME 05-JUN-1988 11:55:22	In examples, system output (what the system displays) is shown in black. User input (what you enter) is shown in red.
$ TYPE MYFILE.DAT . . .	In examples, a vertical series of periods, or ellipsis, means either that not all the data that the system would display in response to a command is shown or that not all the data a user would enter is shown.
input-file, . . .	In examples, a horizontal ellipsis indicates that additional parameters, values, or other information can be entered, that preceding items can be repeated one or more times, or that optional arguments in a statement have been omitted.
[logical-name]	Brackets indicate that the enclosed item is optional. (Brackets are not, however, optional in the syntax of a directory name in a file specification or in the syntax of a substring specification in an assignment statement.)

Convention	Meaning
quotation marks apostrophes	The term quotation marks is used to refer to double quotation marks ("). The term apostrophe (') is used to refer to a single quotation mark.

New and Changed Features

The DIGITAL Command Language has been enhanced for VMS Version 5.0. The following "General Notes" section lists general changes, as well as changes in the behavior of existing commands. Subsequent sections list new commands and new language constructs.

General Notes

Support for the VT300 terminal has been added to the SET TERMINAL command. The VT300 terminal characteristics are the same as those for a VT200 terminal, except that bit CRT-3 is set.

The IF command has been enhanced to accept multiple statements for execution upon a true condition. The IF command has also been enhanced to accept an optional ELSE statement for execution when the condition specified to IF is false.

New DCL Commands and Lexical Functions

The following new DCL commands, lexical functions, and utilities have been added since the Version 4.4 edition of the *VMS DCL Dictionary*:

F$GETQUI
INSTALL
NCS (National Character Set) Utility
SET HOST/DUP
SHOW ENTRY
SUBROUTINE
ENDSUBROUTINE

Changed and Superseded DCL Command and Qualifier Names

The SET CLUSTER/QUORUM command has been superseded by the SET CLUSTER/EXPECTED_VOTES command. Like SET CLUSTER/QUORUM, SET CLUSTER/EXPECTED_VOTES allows users to adjust the cluster quorum dynamically.

The following DCL commands have been superseded by the SET ACL/OBJECT_TYPE=type command qualifier:

SET DEVICE/ACL
SET DIRECTORY/ACL
SET FILE/ACL

See the description of the SET ACL/OBJECT_TYPE=type command qualifier for more information.

The SET QUEUE/ENTRY command has been superseded by the SET ENTRY command. See the description of the SET ENTRY command for more information.

New Qualifiers and Keywords for Existing DCL Commands

The following command qualifiers and keywords have been added since the Version 4.4 edition of the *VMS DCL Dictionary*:

EDIT/TPU	/INITIALIZATION=file-spec
	/START_POSITION=n,m
INITIALIZE/QUEUE	/CLOSE
	/DESCRIPTION
	/DEVICE
	/OPEN
RECALL	/ERASE
SEARCH	/BACKUP
	/BEFORE=time
	/BY_OWNER=uic
	/CONFIRM
	/CREATED
	/EXPIRED
	/FORMAT=NOFF
	/MODIFIED
	/SINCE=time
SET AUDIT	/FAILURE_MODE=option
SET FILE	/STATISTICS
SET CLUSTER	/QUORUM
SET HOST	/BUFFER_SIZE=n
	/[NO]RESTORE
SET PROCESS	/SUSPEND=SUPERVISOR
	/SUSPEND=KERNEL
SET QUEUE	/CLOSE
	/DESCRIPTION
	/OPEN
SHOW AUDIT	/FAILURE_MODE
	/ALL
SHOW QUEUE	/BY_JOB_STATUS
	/SUMMARY
START/QUEUE	/CLOSE
	/DESCRIPTION
	/OPEN

Deleted DCL Command Qualifiers

The SET PROCESS/CPU=[NO]ATTACHED qualifier has been deleted for VMS Version 5.0.

= (Assignment Statement)

Defines a symbolic name for a character string or integer value.

FORMAT

symbol-name =[=] expression

symbol-name[bit-position,size] =[=] replacement-expression

PARAMETERS

symbol-name

Specifies a 1 to 255 character alphanumeric string name for the symbol. The name can contain any alphanumeric characters from the DEC Multinational Character Set, the underscore (_), and the dollar sign ($). However, the name must begin *only* with an alphabetic character (uppercase and lowercase characters are equivalent), an underscore, or a dollar sign. Using one equal sign (=) places the symbol name in the local symbol table for the current command level. Using two equal signs (==) places the symbol name in the global symbol table.

expression

Names the value on the right-hand side of an assignment statement. Can consist of a character string, an integer, a symbol name, a lexical function, or a combination of these entities. The components of the expression are evaluated, and the result is assigned to the symbol. All literal character strings must be enclosed in quotation marks. If the expression contains a symbol, the expression is evaluated using the symbol's value.

The result of expression evaluation is either a character string or a signed integer value. If the expression is evaluated as a string, the symbol is assigned a string value. If the expression is evaluated as an integer, the symbol is assigned an integer value. If the integer value exceeds the capacity of the four byte buffer that holds it, no error message is issued.

For a summary of operators used in expressions, details on how to specify expressions, and details on how expressions are evaluated, see Chapter 5 of the *VMS DCL Concepts Manual*.

DCL uses a buffer that is 1024 bytes long to hold an assignment statement, and to evaluate the expression. The length of the symbol name, the expression, and the expression's calculations cannot exceed 1024 bytes.

[bit-position,size]

States that a binary overlay is to be inserted in the current 32-bit value of a symbol-name. The current value of the symbol-name is evaluated. Then, the specified number of bits is replaced by the result of the replacement-expression. The bit-position is the location relative to bit 0 at which the overlay is to occur. If the symbol you are overlaying is an integer, then the bit position must be less than 32. The sum of the bit position and the size must be less than or equal to 32.

If the symbol you are overlaying is a string, then the bit position must be less than 6152. Because each character is represented using 8 bits, you can begin an overlay at any character through the 768th. (The 768th character starts in bit position 6144.) The sum of the bit position and the size must be less than or equal to 6152.

The size is the number of bits to be overlaid. If you specify a size that is greater than 32, then DCL reduces the size to 32.

The square brackets are required notation; no spaces are allowed between the symbol name and the left bracket. Specify values for bit-position and size as integers.

replacement-expression

Specifies the value that is used to overlay the symbol you are modifying. Specify the replacement-expression as an integer.

If the symbol you are modifying is an integer, the replacement-expression defines a bit pattern that is overlaid on the value assigned to the symbol. If the symbol you are modifying is a character string, the result of the replacement-expression defines a bit pattern that is overlaid on the specified bits of the character string. If the symbol you are modifying is undefined, the result of the replacement-expression is overlaid on a null string.

DESCRIPTION

Symbols defined using assignment statements allow you to extend the command language. At the interactive command level, you can use symbols to define synonyms for commands or command lines. In command procedure files, you can use symbols to provide for conditional execution and substitution of variables.

The maximum number of symbols that can be defined at any time depends on the following:

- The amount of space available to the command interpreter to contain symbol tables and labels for the current process. The amount of space is determined for each process by the SYSGEN parameter CLISYMTBL.
- The size of the symbol names and their values. The command interpreter allocates space for a symbol name and its value. In addition, a few bytes of overhead are allocated for each symbol.

EXAMPLES

1
```
$ LIST == "DIRECTORY"
```

The assignment statement in this example assigns the user-defined synonym LIST as a global symbol definition for the DCL command DIRECTORY.

2

```
$ COUNT = 0
$ LOOP:
$       COUNT = COUNT + 1
$       IF P'COUNT' .EQS. "" THEN EXIT
$       APPEND/NEW &P'COUNT' SAVE.ALL
$       DELETE &P'COUNT';*
$       IF COUNT .LT. 8 THEN GOTO LOOP
$ EXIT
```

This command procedure, COPYDEL.COM, appends files (specified as parameters) to a file called SAVE.ALL. After a file has been appended, the command procedure deletes the file. Up to eight file names can be passed to the command procedure. The file names are assigned to the symbols P1, P2, and so on.

The command procedure uses a counter to refer to parameters that are passed to it. Each time through the loop, the procedure uses an IF command to check whether the value of the current parameter is a null string. When the IF command is scanned, the current value of the symbol COUNT is concatenated with the letter P. The first time through the loop, the IF command tests P1; the second time through the loop it tests P2, and so on. After the expression P'COUNT' is evaluated, the substitution of the file names that correspond to P1, P2, and so on, is automatic within the context of the IF command.

The APPEND and DELETE commands do not automatically perform any substitution, because they expect and require file specifications as input parameters. The ampersand (&) precedes the P'COUNT' expression for these commands to force the appropriate symbol substitution. When these commands are initially scanned each time through the loop, COUNT is substituted with its current value. Then, when the commands execute, the ampersand causes another substitution: the first file specification is substituted for P1, the second file specification is substituted for P2, and so on.

To invoke this procedure, use the following command:

```
$@COPYDEL ALPHA.TXT BETA.DOC
```

The files ALPHA.TXT and BETA.DOC are each appended to the file SAVE.ALL and then deleted.

3

```
$ A = 25
$ CODE = 4 + F$INTEGER("6") - A
$ SHOW SYMBOL CODE
  CODE = -15   HEX = FFFFFFF1   Octal = 1777761
```

This example contains two assignment statements. The first statement assigns the value 25 to the symbol A. The second assignment statement evaluates an expression containing an integer (4), a lexical function (F$INTEGER("6")), and the symbol A. The result of the expression, –15, is assigned to the symbol CODE.

= (Assignment Statement)

4

```
$ FILENAME = "JOBSEARCH" - "JOB"
$ FILETYPE = ".OBJ"
$ FILESPEC = FILENAME + FILETYPE
$ TYPE 'FILESPEC'
```

The first command in this example assigns the symbol FILENAME the value "SEARCH". Notice that the string "SEARCH" is the result of the string reduction operation performed by the expression. The second command assigns the symbol FILETYPE the character string ".OBJ". The symbols FILENAME and FILETYPE are then added together in an expression assigned to the symbol FILESPEC. Because the values of the symbols FILENAME and FILETYPE are concatenated, the resultant value assigned to FILESPEC is the character string "SEARCH.OBJ". The symbol FILESPEC is then used as a parameter for the TYPE command. The apostrophes request the command interpreter to replace the symbol FILESPEC with its value SEARCH.OBJ. Thus, the TYPE command types the file named SEARCH.OBJ.

5

```
$ BELL[0,32] = %X07
$ SHOW SYMBOL BELL
  BELL = ""
```

In this example, the symbol BELL is created with an arithmetic overlay assignment statement. Because the symbol BELL is previously undefined, the hexadecimal value 7 is inserted over a null character string and is interpreted as the ASCII code for the bell character on a terminal. When you issue the command SHOW SYMBOL BELL, the terminal beeps.

If the symbol BELL had been previously defined with an integer value, the result of displaying BELL would have been to show its new integer value.

:= (String Assignment)

Defines a symbolic name for a character string value.

FORMAT

symbol-name :=[=] string

symbol-name[offset,size] :=[=] replacement-string

PARAMETERS

symbol-name

Specifies a 1 to 255-character string name for the symbol. The name can contain any alphanumeric characters from the DEC Multinational Character Set, the underscore, and the dollar sign. However, the name must begin *only* with an alphabetic character, an underscore (_), or a dollar sign ($). Using one equal sign (:=) places the symbol name in the local symbol table for the current command level. Using two equal signs (:==) places the symbol name in the global symbol table.

string

Names the character string value to be equated to the symbol. The string can contain any alphanumeric or special characters. DCL uses a buffer that is 1024 bytes long to hold a string assignment statement. Therefore, the length of the symbol name, the string, and any symbol substitution within the string cannot exceed 1024 characters.

With the := string assignment statement, you do not need to enclose a string literal in quotation marks. String values are automatically converted to uppercase. Also, any leading and trailing spaces and tabs are removed, and multiple spaces and tabs between characters are compressed to a single space.

It is easier to use the assignment statement (=) to create symbols with string values because the assignment statement does not automatically convert letters to uppercase and remove extra spaces. Also, the assignment statement allows you to perform string operations in expressions.

To prohibit uppercase conversion and retain required space and tab characters in a string, place quotation marks around the string. To use quotation marks in a string, enclose the entire string in quotation marks and use a double set of quotation marks within the string. For example:

```
$ TEST := "this     is a ""test"" string"
$ SHOW SYMBOL TEST
  TEST = "this     is a "test" string"
```

In this example, the spaces, lowercase letters, and quotation marks are preserved in the symbol definition.

To continue a symbol assignment on more than one line, use the hyphen as a continuation character. For example:

```
$ LONG_STRING := THIS_IS_A_VERY_LONG-
_$ _SYMBOL_STRING
```

:= (String Assignment)

To assign a null string to a symbol using the string assignment statement, do not specify a string. For example:

```
$ NULL :=
```

Specify the string as a string literal, or as a symbol or lexical function that evaluates to a string literal. If you use symbols or lexical functions, place apostrophes around them to request symbol substitution. See Chapter 7 of the *VMS DCL Concepts Manual* for more information on symbol substitution.

You can also use the string assignment statement to define a foreign command. See Section 5.4 of the *VMS DCL Concepts Manual* for more information about foreign commands.

[offset,size]

Specifies that a portion of a symbol value is to be overlaid with a replacement string. This form of the string assignment statement evaluates the value assigned to a symbol and then replaces the portion of the value (defined by the offset and size) with the replacement string. The square brackets are required notation, and no spaces are allowed between the symbol name and the left bracket.

The offset specifies the character position relative to the beginning of the symbol name's string value at which replacement is to begin. Offset values start at 0.

If the offset is greater than the offset of the last character in the string you are modifying, spaces are inserted between the end of the string and the offset where the replacement string is added. The maximum offset value you can specify is 768.

The size specifies the number of characters to replace. Size values start at 1.

Specify the offset and size as integer expressions. See Section 6.2 of the *VMS DCL Concepts Manual* for more information on integer expressions. The value of the size plus the offset must not exceed 769.

replacement-string

Specifies the string that is used to overwrite the string you are modifying. If the replacement-string is shorter than the size argument, the replacement string is blank-filled on the right until it equals the specified size. Then the replacement string overwrites the string assigned to the symbol name. If the replacement string is longer than the size argument, then the replacement string is truncated on the right to the specified size.

You can specify the replacement-string as a string literal, or as a symbol or lexical function that evaluates to a string literal. If you use symbols or lexical functions, place apostrophes around them to request symbol substitution. See Chapter 7 of the *VMS DCL Concepts Manual* for more information on symbol substitution.

EXAMPLES

1

```
$ TIME := SHOW TIME
$ TIME
  15-APR-1988 11:55:44
```

In this example, the symbol TIME is equated to the command string SHOW TIME. Because the symbol name appears as the first word in a command string, the command interpreter automatically substitutes it with its string value and executes the command SHOW TIME.

2

```
$ STAT := $DBA1:[CRAMER]STAT
$ STAT
```

This example shows how to define STAT as a foreign command. The symbol STAT is equated to a string that begins with a dollar sign followed by a file specification. The command interpreter assumes that the file specification is that of an executable image, that is, a file with a file type of EXE. The symbol STAT in this example becomes a synonym for the following command:

```
$ RUN DBA1:[CRAMER]STAT.EXE
```

When you subsequently type STAT, the command interpreter executes the image.

3

```
$ A = "this is a big     space."
$ SHOW SYMBOL A
  A = "this is a big     space."
$ B := 'A'
$ SHOW SYMBOL B
  B = "THIS IS A BIG SPACE."
```

This example compares the assignment and the string assignment statements. The symbol A is defined using the assignment statement, so lowercase letters and multiple spaces are retained. The symbol B is defined using the string assignment statement. Note that the apostrophes are required; otherwise, the symbol name B would have been equated to the literal string A. However, when symbol A's value is assigned to symbol B, the letters are converted to uppercase and multiple spaces are compressed.

4

```
$ FILE_NAME := MYFILE
$ FILE_NAME[0,2]:= OL
$ SHOW SYMBOL FILE_NAME
  FILE_NAME = "OLFILE"
```

In this example, the substring expression in the assignment statement overlays the first two characters of the string assigned to the symbol FILE_NAME with the letters OL. The offset of 0 requests that the overlay begin with the first character in the string, and the size specification of 2 indicates the number of characters to overlay.

:= (String Assignment)

5

```
$ FILE_NAME := MYFILE
$ FILE_TYPE := .TST
$ FILE_NAME[F$LENGTH(FILE_NAME),4] := 'FILE_TYPE'
$ SHOW SYMBOL FILE_NAME
  FILE_NAME = "MYFILE.TST"
```

In this example, the symbol name FILE_NAME is equated to the string MYFILE and the symbol name FILE_TYPE is equated to the string .TST. The third assignment statement uses the lexical function F$LENGTH to define the offset value where the overlay is to begin. The symbol name FILE_TYPE is used to refer to the replacement string (.TST). Note that you must use apostrophes to request symbol substitution.

The F$LENGTH lexical function returns the length of the string equated to the symbol FILE_NAME; this length is used as the offset. The expression requests that four characters of the string currently equated to the symbol FILE_TYPE be placed at the end of the string currently equated to FILE_NAME. The resultant value of the symbol FILE_NAME is MYFILE.TST.

@ (Execute Procedure)

Executes a command procedure or requests the command interpreter to read subsequent command input from a specific file or device.

FORMAT

@ *file-spec [p1 [p2 [... p8]]]*

PARAMETERS

file-spec

Specifies either the input device or file for the preceding command, or the command procedure to be executed. The default file type is COM. Wildcard characters are not allowed in the file specification.

p1 [p2 [... p8]]

Specifies from one to eight optional parameters to pass to the command procedure. The symbols (P1, P2, . . . P8) are assigned character string values in the order of entry. The symbols are local to the specified command procedure. Separate each parameter with one or more blanks. Use two consecutive quotation marks ("") to specify a null parameter. You can specify a parameter with a character string value containing alphanumeric or special characters, with the following restrictions:

- The command interpreter converts alphabetic characters to uppercase and uses blanks to delimit each parameter. To pass a parameter that contains embedded blanks or literal lowercase letters, place the parameter in quotation marks.
- If the first parameter begins with a slash character (/), you must enclose the parameter in quotation marks.
- To pass a parameter that contains literal quotation marks and spaces, enclose the entire string in quotation marks and use two quotation marks within the string. For example, the command procedure TEST.COM contains the following line:

```
$ WRITE SYS$OUTPUT P1
```

Enter the following at the DCL prompt ($):

```
$ @TEST "Never say ""quit"""
```

When the procedure TEST.COM executes, the parameter P1 is equated to the following string:

```
Never say "quit"
```

If a string contains quotation marks and does not contain spaces, the quotation marks are preserved in the string and the letters within the quotation marks remain in lowercase. For example, enter the following at the DCL prompt:

```
$ @TEST abc"def"ghi
```

When the procedure TEST.COM executes, the parameter P1 is equated to the following string:

```
ABC"def"GHI
```

To use a symbol as a parameter, enclose the symbol in apostrophes to force symbol substitution. For example:

```
$ NAME = "JOHNSON"
$ @INFO 'NAME'
```

The apostrophes cause the value "JOHNSON" to be substituted for the symbol NAME. Therefore, the parameter "JOHNSON" is passed as P1 to INFO.COM.

DESCRIPTION

Use the @ command to execute a command procedure containing

- DCL command lines and/or data
- Qualifiers and/or parameters for a specific command line

To execute a command procedure containing commands or data, or both, place the @ command at the beginning of a command line and then specify the name of the command procedure file. The command procedure can contain DCL commands and input data for a command or program that is currently executing. All DCL commands in a command procedure must begin with a dollar sign ($) character. If a command is continued with the continuation character (-), the subsequent lines must not begin with a dollar sign.

Any line in a command procedure that does not contain a dollar sign in the first character position (and is not a continuation line) is treated as input data for the command or program that is currently executing. The DECK command allows you to specify that data contains dollar signs in record position one.

A command procedure can also contain the @ command to execute another command procedure. The maximum command level you can achieve by nesting command procedures is sixteen, including the top-level command procedure. Command procedures can also be queued for processing as batch jobs, either by using the SUBMIT command or by placing a deck of cards containing the command procedure in the system card reader.

To execute a command procedure that contains qualifiers or parameters, or both, for a specific command line, place the @ command where the qualifiers or parameters normally would be in the command line. Then specify the name of the command procedure file containing the qualifiers or parameters.

If the command procedure file begins with *parameters* for the command, the @ command must be preceded by a space. For example,

```
$ CREATE TEST.COM
TIME    CTRL/Z
$ SHOW @TEST
  09-DEC-1988 17:20:26
```

If the file begins with *qualifiers* for the command, do *not* precede the @ command with a space. For example,

```
$ CREATE TEST_2.COM
/SIZE    CTRL/Z
$ DIR@TEST_2

Directory WORK$:[SCHEDULE]

JANUARY.TXT;8          14-AUG-1988 15:47:45.57
FEBRUARY.TXT;7         14-AUG-1988 15:43:16.20
MARCH.TXT;6            12-AUG-1988 11:11:45.74

Total of 3 files.
```

If the file contains parameters, or qualifiers, or both, do *not* begin the lines in the file with dollar signs ($). Any additional data on the command line following @file-spec is treated as parameters for the procedure.

QUALIFIER

/OUTPUT=file-spec

The name of the file to which the command procedure output is written. By default, the output is written to the current SYS$OUTPUT device. The default output file type is LIS. Wildcard characters are not allowed in the output file specification. System responses and error messages are written to SYS$COMMAND as well as to the specified file. The /OUTPUT qualifier must immediately follow the file specification of the command procedure; otherwise, the qualifier is interpreted as a parameter to pass to the command procedure.

You can also redefine SYS$OUTPUT to redirect the output from a command procedure. If you place the following command as the first line in a command procedure, output will be directed to the file you specify:

```
$ DEFINE SYS$OUTPUT file-spec
```

When the procedure exits, SYS$OUTPUT will be restored to its original equivalence string. This produces the same result as using the /OUTPUT qualifier when you execute the command procedure.

EXAMPLES

1

```
$ CREATE DOFOR.COM
$ ON WARNING THEN EXIT
$ IF P1.EQS."" THEN INQUIRE P1 FILE
$ FORTRAN/LIST 'P1'
$ LINK 'P1'
$ RUN 'P1'
$ PRINT 'P1'
CTRL/Z
$ @DOFOR AVERAGE
```

This example shows a command procedure, named DOFOR.COM, that executes the FORTRAN, LINK, and RUN commands to compile, link, and execute a program. The ON command requests that the procedure not continue if any of the commands result in warnings or errors.

When you execute DOFOR.COM, you can pass the file specification of the FORTRAN program as the parameter P1. If you do not specify a value for P1 when you execute the procedure, the INQUIRE command issues a prompting message to the terminal and equates what you enter with the symbol P1. In this example, the file name AVERAGE is assigned to P1. The file type is not included because the commands FORTRAN, LINK, RUN, and PRINT provide default file types.

2
```
$ @MASTER/OUTPUT=MASTER.LOG
```

This command executes a procedure named MASTER.COM; all output is written to the file MASTER.LOG.

3
```
$ CREATE FILES.COM
*.FOR, *.OBJ
CTRL/Z
$ DIRECTORY @FILES
```

This example shows a command procedure, FILES.COM, that contains parameters for a DCL command line. You can execute this procedure after the DIRECTORY command to get a listing of all FORTRAN source and object files in your current default directory.

4
```
$ CREATE QUALIFIERS.COM
/DEBUG/SYMBOL_TABLE/MAP/FULL/CROSS_REFERENCE
CTRL/Z
$ LINK SYNAPSE@QUALIFIERS
```

This example shows a command procedure, QUALIFIERS.COM, that contains qualifiers for the LINK command. When you enter the LINK command, specify the command procedure immediately after the file specification of the file you are linking. Do not type a space between the file specification and the @ command.

5
```
$ CREATE SUBPROCES.COM
$ RUN 'P1' -
  /BUFFER_LIMIT=1024 -
  /FILE_LIMIT=4 -
  /PAGE_FILES=256 -
  /QUEUE_LIMIT=2 -
  /SUBPROCESS_LIMIT=2 -
  'P2'  'P3'  'P4'  'P5'  'P6'  'P7'  'P8'
CTRL/Z
$ @SUBPROCES  LIBRA  /PROCESS_NAME=LIBRA
```

This example shows a command procedure named SUBPROCES.COM. This procedure issues the RUN command to create a subprocess to execute an image and also contains qualifiers defining quotas for subprocess creation. The name of the image to be run is passed as the parameter P1. P2 through P8 can be used to specify additional qualifiers.

In this example, the file name LIBRA is equated to P1; it is the name of an image to execute in the subprocess. The qualifier /PROCESS_NAME=LIBRA is equated to P2; it is an additional qualifier for the RUN command.

6

```
$ CREATE EDOC.COM
$ ASSIGN SYS$COMMAND:  SYS$INPUT
$ NEXT:
$       INQUIRE NAME "File name"
$       IF NAME.EQS."" THEN EXIT
$       EDIT/EDT 'NAME'.DOC
$       GOTO NEXT
CTRL/Z
$ @EDOC
```

This procedure, named EDOC.COM, invokes the EDT editor. When an edit session is terminated, the procedure loops to the label NEXT. Each time through the loop, the procedure requests another file name for the editor and supplies the default file type of DOC. When a null line is entered in response to the INQUIRE command, the procedure terminates with the EXIT command.

The ASSIGN command changes the equivalence name of SYS$INPUT for the duration of the procedure. This change allows the EDT editor to read input data from the terminal, rather than from the command procedure file (the default input data stream if SYS$INPUT had not been changed). When the command procedure exits, SYS$INPUT is reassigned to its original value.

ACCOUNTING

Invokes the Accounting Utility to collect, record, and report accounting data. For a complete description of the Accounting Utility, including information about the ACCOUNTING command, refer to the *VMS Accounting Utility Manual*.

FORMAT ACCOUNTING *file-spec[,...]*

ALLOCATE

Provides your process with exclusive access to a device until you deallocate the device or terminate your process. Optionally associates a logical name with the device.

FORMAT

ALLOCATE *device-name[:][,...] [logical-name[:]]*

PARAMETERS

device-name[:][,...]

Specifies the name of a physical device or a logical name that translates to the name of a physical device. The device name can be generic: if no controller or unit number is specified, any device that satisfies the specified part of the name is allocated. If more than one device is specified, the first available device is allocated.

logical-name

Specifies a character string of 1 through 255 characters. Enclose the string in quotation marks (") if it contains blanks. Trailing colons are not used. The name becomes a process logical name with the device name as the equivalence name. The logical name remains defined until it is explicitly deleted or your process terminates.

QUALIFIERS

/GENERIC
/NOGENERIC (default)

Indicates that the first parameter is a device *type* rather than a device *name*. Example device types are RX50, RD52, TK50, RC25, RCF25, RL02. The first free, nonallocated device of the specified name and type is allocated.

The /[NO]GENERIC qualifier is placed before the device-name parameter in the ALLOCATE command line. For example, you can allocate an RK07 device by entering the following command at the DCL prompt ($):

```
$ ALLOCATE/GENERIC RK07
```

The following table shows some device types that you can specify with the /GENERIC qualifier:

"D" DEVICES	"T" DEVICES
RA60/70/80/81 /90	TA78/79/81
RC25/RCF25	TK50/70
RK06/7	TS11
RL01/2	TU16
RM03/05/80	TU58
RP04/5/6/7	TU77/78/79/80/81
RX01/2/4/33	

/LOG (default)
/NOLOG

Displays a message indicating the name of the device allocated. If the operation specifies a logical name that is currently assigned to another device, displays the superseded value.

EXAMPLES

1
```
$ ALLOCATE  DMB2:
%DCL-I-ALLOC, DMB2: allocated
```

The ALLOCATE command in this example requests the allocation of a specific RK06/RK07 disk drive, that is, unit 2 on controller B. The system response indicates that the device was successfully allocated.

2
```
$ ALLOCATE  MT,MF:   TAPE:
%DCL-I-ALLOC, MTB2: allocated
.
.
.
$ SHOW LOGICAL TAPE:
TAPE: = _MTB2:    (process)
$ DEALLOCATE TAPE:
$ DEASSIGN TAPE:
```

The ALLOCATE command in this example requests the allocation of any tape device whose name begins with MT or MF, to be assigned the logical name TAPE. The ALLOCATE command locates an available tape device whose name begins with MT, and responds with the name of the device allocated. (If no tape device beginning with MT had been found, the ALLOCATE command would have searched for a device beginning with MF.) Subsequent references to the device TAPE in user programs or command strings are translated to the device name MTB2.

When the tape device is no longer needed, the DEALLOCATE command deallocates it and the DEASSIGN command deletes the logical name. Note that the logical name TAPE was specified with a colon on the ALLOCATE command, but that the logical name table entry does not have a colon.

```
3  $ ALLOCATE/GENERIC RL02 WORK
   %DCL-I-ALLOC, _DLA1: allocated
   %DCL-I-SUPERSEDE, previous value of WORK has been superseded
```

The ALLOCATE command in this example requests the allocation of any RL02 disk device and assigns the logical name WORK to the device. The completion message identifies the allocated device and indicates that the assignment of the logical name WORK supersedes a previous assignment of that name.

```
4  $ ALLOCATE $TAPE1
   %DCL-I-ALLOC, _MUA0: allocated
```

The ALLOCATE command in this example allocates the tape device _MUA0, which is associated with the logical name $TAPE1.

```
5  $ ALLOCATE /GENERIC RX50 ACCOUNTS
```

The ALLOCATE command in this example allocates the first free floppy disk drive and makes its name equivalent to the process logical name ACCOUNTS.

ANALYZE/CRASH_DUMP

Invokes the System Dump Analyzer Utility (SDA) for analysis of a system dump file. The /CRASH_DUMP qualifier is required. For a complete description of the System Dump Analyzer Utility, including more information about the ANALYZE/CRASH_DUMP command and its qualifier, see the *VMS System Dump Analyzer Utility Manual*.

FORMAT

ANALYZE/CRASH_DUMP *file-spec*

ANALYZE/DISK_STRUCTURE

Invokes the Analyze/Disk_Structure Utility to do the following:

- Check the readability and validity of Files–11 Structure Level 1 and Files–11 Structure Level 2 disk volumes
- Report errors and inconsistencies

The /DISK_STRUCTURE qualifier is required. For a complete description of the Analyze/Disk_Structure Utility, including information about the DCL command ANALYZE/DISK_STRUCTURE and its qualifiers, see the *VMS Analyze/Disk_Structure Utility Manual.*

FORMAT

ANALYZE/DISK_STRUCTURE *device-name:*

ANALYZE/ERROR_LOG

Invokes the Errorlog Report Formatter (ERF) to report selectively the contents of an error log file. The /ERROR_LOG qualifier is required. For a complete description of the Error Log Utility, including more information about the ANALYZE/ERROR_LOG command and its qualifiers, see the *VMS Error Log Utility Manual*.

FORMAT **ANALYZE/ERROR_LOG** *[/qualifier(s)] [file-spec[,...]]*

ANALYZE/IMAGE

Analyzes the contents of an executable image file or a shareable image file and checks for obvious errors in the image file. See the description of the linker in the *VMS Linker Utility Manual* for general information about image files. The /IMAGE qualifier is required. (Use the ANALYZE/OBJECT command to analyze the contents of an object file.)

FORMAT

ANALYZE/IMAGE *file-spec [,...]*

PARAMETER

file-spec[,...]

Specifies the image files you want analyzed (default file type is EXE.) Use commas or plus signs to separate file specifications. Wildcard characters are allowed.

DESCRIPTION

The ANALYZE/IMAGE command provides a description of the components of an executable image file or shareable image file. It also verifies that the structure of the major parts of the image file is correct. However, the ANALYZE/IMAGE command cannot ensure that program execution is error-free.

If an error is found, the first error of the worst severity that is discovered is returned. For example, if a warning (A) and two errors (B and C) are signaled, then the first error (B) is returned as the image exit status, which is placed in the DCL symbol $STATUS at image exit.

The ANALYZE/IMAGE command provides the following information:

- Image type—Identifies whether the image is executable or shareable.
- Image transfer addresses—Identify the addresses to which control is passed at image execution time.
- Image version—Identifies the revision level of the image.
- Patch information—Indicates whether the image has been patched (changed without having been recompiled or reassembled and relinked). If a patch is present, the actual patch code can be displayed.
- Location of the Debug Symbol Table (DST)—Identifies the location of the DST in the image file. DST information is present only in executable images that have been linked with the /DEBUG or /TRACEBACK command qualifiers.
- Location of the global symbol table (GST)—Identifies the location of the GST in the image file. GST information is present only in shareable image files.
- Image section descriptors (ISD)—Identifies portions of the image binary contents that are grouped in clusters according to their attributes. An ISD contains information that the image activator needs when it initializes the address space for an image. For example, it tells whether or not the ISD is shareable, if it is readable or writable, if it is based or position-independent, and how much memory should be allocated.

- Fixup vectors—Contain information that the image activator needs to ensure the position-independence of shareable image references.

The ANALYZE/IMAGE command has command qualifiers and positional qualifiers. By default, if you do not specify any positional qualifiers (for example, /GST or /HEADER), the entire image is analyzed. If you do specify a positional qualifier, the analysis excludes all other positional qualifiers except for the /HEADER qualifier (which is always enabled) and those you explicitly request.

QUALIFIERS

/FIXUP_SECTION
Positional Qualifier.

Specifies that the analysis should include all information in the fixup section of the image.

If you want the analysis to include the fixup section of all image files in the parameter list, insert the /FIX_UP qualifier immediately following the /IMAGE qualifier.

If you want the analysis to include fixup sections selectively, insert the /FIX_UP qualifier immediately following the selected file specification(s).

/GST
Positional Qualifier.

Specifies that the analysis should include all global symbol table records. This qualifier is valid only for shareable images.

If you want the analysis to include the global symbol table records of all image files in the parameter list, insert the /GST qualifier immediately following the /IMAGE qualifier.

If you want the analysis to include global symbol table records selectively, insert the /GST qualifier immediately following the selected file specification(s).

/HEADER
Positional Qualifier.

Specifies that the analysis should include only header items and image section descriptions, unless the command explicitly specifies other information. The image header items are always analyzed.

/INTERACTIVE
/NOINTERACTIVE (default)
Specifies whether or not the analysis is interactive. In interactive mode, as each item is analyzed, the results are displayed on the screen and you are asked whether you want to continue.

/OUTPUT=file-spec
Directs the output of the image analysis (default is SYS$OUTPUT.) No wildcard characters are allowed. If you specify a file type and omit the file name, the default file name ANALYZE is used. The default file type is ANL.

/PATCH_TEXT

Positional Qualifier.

Specifies that the analysis include all patch text records. If you want the analysis to include the patch text records for each image file in the parameter list, insert the /PATCH_TEXT qualifier immediately following the /IMAGE qualifier.

If you want the analysis to include patch text records selectively, insert the /PATCH_TEXT qualifier immediately following the selected file specification(s).

EXAMPLES

1

```
$ ANALYZE/IMAGE  LINEDT
```

The ANALYZE/IMAGE command in this example produces a description and an error analysis of the image LINEDT.EXE. Output is directed to the current SYS$OUTPUT device. By default, the entire image is analyzed.

2

```
$ ANALYZE/IMAGE/OUTPUT=LIALPHEX/FIXUP_SECTION/PATCH_TEXT  LINEDT, ALPHA
```

The ANALYZE/IMAGE command in this example stores a description and an error analysis of the fixup sections and patch text records of LINEDT.EXE and ALPHA.EXE in file LIALPHEX.ANL. The output is directed to the file LIALPHEX.ANL.

ANALYZE/MEDIA

Invokes the Bad Block Locator Utility (BAD), which analyzes block-addressable devices and records the location of blocks that cannot reliably store data. For a complete description of the Bad Block Locator Utility, including information about the ANALYZE/MEDIA command and its qualifiers, see the *VMS Bad Block Locator Utility Manual*.

FORMAT ANALYZE/MEDIA *device*

ANALYZE/OBJECT

Analyzes the contents of an object file and checks for any obvious errors. The /OBJECT qualifier is required. (The ANALYZE/IMAGE command analyzes the contents of an image file.)

FORMAT

ANALYZE/OBJECT *file-spec[,...]*

PARAMETER

file-spec[,...]

Specifies the object files or object module libraries you want analyzed (default file type is OBJ). Use commas or plus signs to separate file specifications. Wildcard characters are allowed.

DESCRIPTION

The ANALYZE/OBJECT command describes the contents of one or more object modules contained in one or more files. It also performs a partial error analysis. This analysis determines whether the records in an object module conform in content, format, and sequence to the specifications of the VMS Object Language.

ANALYZE/OBJECT is intended primarily for programmers of compilers, debuggers, or other software involving VMS object modules. It checks that the object language records generated by the object modules are acceptable to the VMS Linker, and it identifies certain errors in the file. It also provides a description of the records in the object file or object module library. For more information on the VMS linker and on the VMS Object Language, refer to the *VMS Linker Utility Manual.*

The ANALYZE/OBJECT command analyzes the object modules in order, record by record, from the first to the last record in the object module. Fields in each record are analyzed in order from the first to the last field in the record. After the object module is analyzed, you should compare the content and format of each type of record to the required content and format of that record as described by the VMS Object Language. This comparison is particularly important if the analysis output contains a diagnostic message.

The linking of an object module differs from the analysis of an object module. Object language commands are not executed in an analysis, but they are executed in a linking operation. As a result, even if the analysis is error-free, the linking operation may not be. In particular, the analysis does not detect the following:

- That data arguments in TIR commands are in the correct format
- That "Store Data" TIR commands are storing within legal address limits

Therefore, as a final check, you should still link an object module whose analysis is error-free before you assume it is correct.

If an error is found, however, the first error of the worst severity that is discovered is returned. For example, if a warning (A) and two errors (B and C) are signaled, then the first error (B) is returned as the image exit status, which is placed in the DCL symbol $STATUS at image exit.

ANALYZE/OBJECT

ANALYZE/OBJECT uses positional qualifiers; that is, qualifiers whose function depends on their position in the command line. When a positional qualifier precedes all of the input files in a command line, it affects all input files. For example, the following command line requests that the analysis include the global symbol directory records in files A, B, and C:

```
$ ANALYZE/OBJECT/GSD A,B,C
```

Conversely, when a positional qualifier is associated with only one file in the parameter list, only that file is affected. For example, the following command line requests that the analysis include the global symbol directory records in file B only:

```
$ ANALYZE/OBJECT A,B/GSD,C
```

Typically, all records in an object module are analyzed. However, when any of the qualifiers /DBG, /EOM, /GSD, /LNK, /MHD, /TBT, or /TIR are specified, only the record types indicated by the qualifiers are analyzed. All other record types are ignored.

By default, the analysis includes all record types unless you explicitly request a limited analysis using appropriate qualifiers.

Note: End-of-module (EOM) records and module header (MHD) records are always analyzed, no matter which qualifiers you specify.

QUALIFIERS

/DBG

Positional qualifier.

Specifies that the analysis should include all debugger information records. If you want the analysis to include debugger information for all files in the parameter list, insert the /DBG qualifier immediately following the /OBJECT qualifier. If you want the analysis to include debugger information selectively, insert the /DBG qualifier immediately following the selected file specification(s).

/EOM

Positional qualifier.

Specifies that the analysis should be limited to MHD records, EOM records, and records explicitly specified by the command. If you want this to apply to all files in the parameter list, insert the /EOM qualifier immediately following the /OBJECT qualifier.

To make this applicable selectively, insert the /EOM qualifier immediately following the selected file specification(s).

Note: End-of-module records may be EOM or EOMW records. See the *VMS Linker Utility Manual* for more information.

/GSD

Positional qualifier.

Specifies that the analysis should include all global symbol directory records.

If you want the analysis to include global symbol directory records for each file in the parameter list, specify /GSD immediately following the /OBJECT qualifier.

If you want the analysis to include global symbol directory records selectively, insert the /GSD qualifier immediately following the selected file specification(s).

/INCLUDE[=(module[,...])]

When the specified file is an object module library, use this qualifier to list selected object modules within the library for analysis. If you omit the list or specify an asterisk, all modules are analyzed. If you specify only one module, you may omit the parentheses.

/INTERACTIVE
/NOINTERACTIVE (default)

Controls whether the analysis occurs interactively. In interactive mode, as each record is analyzed, the results are displayed on the screen, and you are asked whether you want to continue.

/LNK

Positional qualifier.

Specifies that the analysis should include all link option specification records.

If you want the analysis to include link option specification records for each file in the parameter list, specify /LNK immediately following the /OBJECT qualifier.

If you want the analysis to include link option specification records selectively, insert the /LNK qualifier immediately following the selected file specification(s).

/MHD

Positional qualifier.

Specifies that the analysis should be limited to MHD records, EOM records, and records explicitly specified by the command. If you want this to apply to all files in the parameter list, insert the /MHD qualifier immediately following the /OBJECT qualifier.

To make this applicable selectively, insert the /MHD qualifier immediately following the selected file specification(s).

/OUTPUT[=file-spec]

Directs the output of the object analysis (default is SYS$OUTPUT). If you specify a file type and omit the file name, the default file name ANALYZE is used. The default file type is ANL.

No wildcard characters are allowed in the file specification.

/TBT

Positional qualifier.

Specifies that the analysis should include all module traceback records.

If you want the analysis to include traceback records for each file in the parameter list, specify /TBT immediately following the /OBJECT qualifier.

If you want the analysis to include traceback records selectively, insert the /TBT qualifier immediately following the selected file specification(s).

/TIR

Positional qualifier.

Specifies that the analysis should include all text information and relocation records.

If you want the analysis to include text information and relocation records for each file in the parameter list, specify /TIR immediately following the /OBJECT qualifier.

If you want the analysis to include text information and relocation records selectively, insert the /TIR qualifier immediately following the selected file specification(s).

EXAMPLES

1 `$ ANALYZE/OBJECT/INTERACTIVE LINEDT`

In this example, the ANALYZE/OBJECT command produces a description and a partial error analysis of the object file LINEDT.OBJ. By default, all types of records are analyzed. Output is to the terminal, because the /INTERACTIVE qualifier has been used. As each item is analyzed, the utility displays the results on the screen and asks if you want to continue.

2 `$ ANALYZE/OBJECT/OUTPUT=LIOBJ/DBG LINEDT`

In this example, the ANALYZE/OBJECT command analyzes only the debugger information records of the file LINEDT.OBJ. Output is to the file LIOBJ.ANL.

ANALYZE/PROCESS_DUMP

Invokes the VMS Debugger for analysis of a process dump file that was created when an image failed during execution (use the /DUMP qualifier with the RUN or SET PROCESS commands to generate a dump file). For a complete description of the debugger, including information about the DEBUG command, refer to the *VMS Debugger Manual*.

Requires read (R) access to the dump file.

FORMAT

ANALYZE/PROCESS_DUMP *dump-file*

PARAMETER

dump-file

Specifies the dump file to be analyzed with the debugger.

DESCRIPTION

The ANALYZE/PROCESS_DUMP command examines the dump file of an image that failed during execution. The VMS debugger is invoked automatically. To cause a dump file to be created for a process, you must use the /DUMP qualifier with the RUN command when invoking the image, or you must use the SET PROCESS/DUMP command before invoking the image.

QUALIFIERS

/FULL

Displays all known information about the failing process.

/IMAGE=image-name
/NOIMAGE

Specifies the image whose symbols are to be used in analyzing the dump. If you use the /NOIMAGE qualifier, no symbols are taken from any image. By default, symbols are taken from the image with the same name as the image that was running at the time of the dump.

/INTERACTIVE
/NOINTERACTIVE (default)

Causes the display of information to pause when your terminal screen is filled. Press RETURN to display additional information. By default, the display is continuous.

/MISCELLANEOUS

Displays all the miscellaneous information in the dump.

/OUTPUT=file-spec

Writes the information to the specified file. By default, the information is written to the current SYS$OUTPUT device. No wildcard characters are permitted in the file specification.

/RELOCATION

Displays the addresses to which data structures saved in the dump are mapped in P0 space. (Examples of such data structures are the stacks.) The data structures in the dump must be mapped into P0 so that the debugger can use those data structures in P1 space.

EXAMPLE

```
$ ANALYZE/PROCESS/FULL ZIPLIST

 R0 = 00018292  R1 = 8013DE20  R2 = 7FFE6A40   R3 = 7FFE6A98
 R4 = 8013DE20  R5 = 00000000  R6 = 7FFE7B9A   R7 = 0000F000
 R8 = 00000000  R9 = 00000000  R10 = 00000000  R11 = 00000000
 SP = 7FFAEF44  AP = 7FFAEF48  FP  = 7FFAEF84
 FREE_P0_VA  00001600    FREE_P1_VA  7FFAC600
 Active ASTs  00         Enabled ASTs 0F
 Current Privileges  FFFFFF80  1010C100
 Event Flags  00000000  E0000000
 Buffered I/O count/limit 6/6
 Direct I/O count/limit   6/6
 File count/limit         27/30
 Process count/limit      0/0
 Timer queue count/limit  10/10
 AST count/limit          6/6
 Enqueue count/limit      30/30
 Buffered I/O total 7      Direct I/O total 18
 Link Date  27-DEC-1988 15:02:00.48   Patch Date  17-NOV-1988 00:01:53.71
 ECO Level  0030008C  00540040  00000000  34303230
 Kernel stack 00000000 pages at 00000000 moved to 00000000
 Exec stack 00000000 pages at 00000000 moved to 00000000
 Vector page 00000001 page at 7FFEFE00 moved to 00001600
 PIO (RMS) area 00000005 pages at 7FFE1200 moved to 00001800
 Image activator context 00000001 page at 7FFE3400 moved to 00002200
 User writeable context 0000000A pages at 7FFE1C00 moved to 00002400
Creating a subprocess
        VAX DEBUG Version X5.0-2
DBG>
```

This example shows the output of the ANALYZE/PROCESS command when used with the /FULL qualifier. The file specified, ZIPLIST, contains the dump of a process that encountered a fatal error. The DBG> prompt indicates that the debugger is ready to accept commands.

ANALYZE/RMS_FILE

Invokes the Analyze/RMS_File Utility (ANALYZE/RMS_FILE) to inspect and analyze the internal structure of a VMS RMS file. The /RMS_FILE qualifier is required. For a complete description of the Analyze/RMS_File Utility, including more information about the ANALYZE/RMS_FILE command and its qualifiers, see the *VMS Analyze/RMS_File Utility Manual.*

FORMAT

ANALYZE/RMS_FILE *file-spec[,...]*

ANALYZE/SYSTEM

Invokes the System Dump Analyzer (SDA) for analysis of the running system. The /SYSTEM qualifier is required. For a complete description of the System Dump Analyzer, including more information about the ANALYZE/SYSTEM command and its qualifiers, see the *VMS System Dump Analyzer Utility Manual.*

FORMAT ANALYZE/SYSTEM

APPEND

Adds the contents of one or more specified input files to the end of the specified output file.

FORMAT

APPEND *input-file-spec[,...] output-file-spec*

PARAMETERS

input-file-spec[,...]

Specifies the names of one or more input files to be appended. Multiple input files are appended to the output file in the order specified. If you specify more than one input file, separate multiple file specifications with either commas or plus signs.

You can use wildcard characters in the input file specifications.

output-file-spec

Specifies the name of the file to which the input files will be appended.

You must specify at least one field in the output file specification. If you do not specify a device or directory, the APPEND command uses the current default device and directory. Other unspecified fields default to the corresponding fields of the first input file specification.

If you use the asterisk wildcard character in any fields of the output file specification, the APPEND command uses the corresponding field of the input file specification. If you are appending more than one input file, APPEND uses the corresponding fields from the first input file.

DESCRIPTION

The APPEND command is similar in syntax and function to the COPY command. Normally, the APPEND command adds the contents of one or more files to the end of an existing file without incrementing the version number. The /NEW_VERSION qualifier causes the APPEND command to create a new output file if no file with that name exists.

QUALIFIERS

/ALLOCATION=number-of-blocks

Output-file-spec qualifier.

Forces the initial allocation of the output file to the specified number of 512-byte blocks.

If you do not specify the /ALLOCATION qualifier, the initial allocation of the output file is determined by the size of the input file. The allocation size is applied only if a new file is actually created. Relevant only with the /NEW_VERSION qualifier.

/BACKUP

Modifies the time value specified with the /BEFORE or /SINCE qualifier. /BACKUP selects files according to the dates of their most recent backups. This qualifier is incompatible with the other qualifiers that also allow you to select files according to time attributes: /CREATED, /EXPIRED, and

/MODIFIED. If you specify none of these four time qualifiers, the default is /CREATED.

/BEFORE[=time]

Selects only those files dated prior to the specified time. You can specify time as an absolute time, as a combination of absolute and delta times, or as one of the following keywords: TODAY (default), TOMORROW, or YESTERDAY. Specify one of the following qualifiers with /BEFORE to indicate the time attribute to be used as the basis for selection: /BACKUP, /CREATED (default), /EXPIRED, or /MODIFIED.

See Section 1.4 of the *VMS DCL Concepts Manual* for complete information on specifying time values.

/BY_OWNER[=uic]

Selects only those files whose owner user identification code (UIC) matches the specified owner UIC. The default UIC is that of the current process.

Specify the UIC using standard UIC format as described in Section 8.1 of the *VMS DCL Concepts Manual.*

/CONFIRM
/NOCONFIRM (default)

Controls whether a request is issued before each APPEND operation to confirm that the operation should be performed on that file. The following responses are valid:

YES	NO	QUIT
TRUE	FALSE	CTRL/Z
1	0	ALL
	RET	

You can use any combination of upper- and lowercase letters for word responses. Word responses can be abbreviated to one or more letters (for example, T, TR, or TRU for TRUE), but these abbreviations must be unique. Affirmative answers are YES, TRUE, and 1. Negative answers are NO, FALSE, 0, and the RETURN key. QUIT or CTRL/Z indicates that you want to stop processing the command at that point. When you respond with ALL, the command continues to process, but no further prompts are given. If you type a response other than one of those in the list, DCL issues an error message and redisplays the prompt.

/CONTIGUOUS
/NOCONTIGUOUS

Output-file-spec qualifier.

Specifies that the output file must occupy physically contiguous disk blocks. By default, the APPEND command creates an output file in the same format as the corresponding input file and does not report an error if not enough space exists for a contiguous allocation. Relevant only with the /NEW_VERSION qualifier.

If an input file is contiguous, the APPEND command attempts to create a contiguous output file, but does not report an error if there is not enough space. If you append multiple input files of different formats, the output file may or may not be contiguous. Use the /CONTIGUOUS qualifier to ensure that the output file is contiguous.

/CREATED (default)

Modifies the time value specified with the /BEFORE or /SINCE qualifiers. /CREATED selects files based on their dates of creation. This qualifier is incompatible with the other qualifiers that also allow you to select files according to time attributes: /BACKUP, /EXPIRED, and /MODIFIED. If you specify none of these four time qualifiers, the default is /CREATED.

/EXCLUDE=(file-spec[,...])

Excludes the specified files from the append operation. You can include a directory but not a device in the file specification. Wildcard characters are allowed in the file specification. However, you cannot use relative version numbers to exclude a specific version. If you provide only one file specification, you can omit the parentheses.

/EXPIRED

Modifies the time value specified with the /BEFORE or /SINCE qualifier. /EXPIRED selects files according to their expiration dates. (The expiration date is set with the SET FILE/EXPIRATION_DATE command.) The /EXPIRED qualifier is incompatible with the other qualifiers that also allow you to select files according to time attributes: /BACKUP, /CREATED, and /MODIFIED. If you specify none of these four time qualifiers, the default is /CREATED.

/EXTENSION=number-of-blocks

Output-file-spec qualifier.

Specifies the number of blocks to be added to the output file each time the file is extended. When you specify /EXTENSION, the /NEW_VERSION qualifier is assumed and need not be typed on the command line. Relevant only with the /NEW_VERSION qualifier.

The extension value is applied only if a new file is actually created.

/LOG
/NOLOG (default)

Controls whether the APPEND command displays the file specifications of each file appended. If /LOG is specified, displays the file specifications of the input and output files as well as the number of blocks or records appended after each append operation.

/MODIFIED

Modifies the time value specified with the /BEFORE or /SINCE qualifier. /MODIFIED selects files according to the dates on which they were last modified. This qualifier is incompatible with the other qualifiers that also allow you to select files according to time attributes: /BACKUP, /CREATED, and /EXPIRED. If you specify none of these four time modifiers, the default is /CREATED.

/NEW_VERSION
/NONEW_VERSION (default)

Output-file-spec qualifier.

Controls whether the APPEND command creates a new output file if the specified output file does not exist. (By default, the specified output file already exists.) If the specified output file does not already exist, use the /NEW_VERSION qualifier to create a new output file. If the output file does

exist, the /NEW_VERSION qualifier is ignored and the input file is appended to the output file.

/PROTECTION=(code)

Output-file-spec qualifier.

Specifies protection for the output file. Specify ownership as SYSTEM, OWNER, GROUP, or WORLD and access as R (read), W (write), E (execute), or D (delete). The default protection, including any protection attributes not specified, is that of the existing output file. If no output file exists, the current default protection applies. Relevant only with the /NEW_VERSION qualifier.

See Section 8.1 of the *VMS DCL Concepts Manual* for more information on specifying protection code.

/READ_CHECK
/NOREAD_CHECK (default)

Input-file-spec qualifier.

Reads each record in the input files twice to verify that it has been read correctly.

/SINCE[=time]

Selects for the append operation only those files dated after the specified time. You can specify time as an absolute time, a combination of absolute and delta times, or as one of the following keywords: TODAY (default), TOMORROW, or YESTERDAY. Specify one of the following qualifiers with /BEFORE to indicate the time attribute to be used as the basis for selection: /BACKUP, /CREATED (default), /EXPIRED, or /MODIFIED.

See Section 1.4 of the *VMS DCL Concepts Manual* for complete information on specifying time values.

/WRITE_CHECK
/NOWRITE_CHECK (default)

Output-file-spec qualifier.

Reads each record in the output file after the record is written to verify that it was appended successfully and that the output file can subsequently be read without error.

EXAMPLES

1
```
$ APPEND  TEST3.DAT TESTALL.DAT
```

The APPEND command appends the contents of the file TEST3.DAT from the default disk and directory to the file TESTALL.DAT, also located on the default disk and directory.

2
```
$ APPEND/NEW_VERSION/LOG *.TXT   MEM.SUM
%APPEND-I-CREATED, USE$:[MAL]MEM.SUM;1 created
%APPEND-S-COPIED, USE$:[MAL]A.TXT;2 copied to USE$:[MAL]MEM.SUM;1 (1 block)
%APPEND-S-APPENDED, USE$:[MAL]B.TXT;3 appended to USE$:[MAL]MEM.SUM;1 (3 records)
%APPEND-S-APPENDED, USE$:[MAL]G.TXT;7 appended to USE$:[MAL]MEM.SUM;1 (51 records)
```

The APPEND command appends all files with file types of TXT to a file named MEM.SUM. The /LOG qualifier requests a display of the specifications of each input file appended. If the file MEM.SUM does not exist, the APPEND command creates it, as the output shows. The number of blocks or records shown in the output refers to the source file and not to the target file total.

3
```
$ APPEND/LOG A.DAT, B.MEM   C.*
%APPEND-S-APPENDED, USE$:[MAL]A.DAT;4 appended to USE$:[MAL]C.DAT;4 (2 records)
%APPEND-S-APPENDED, USE$:[MAL]B.MEM;5 appended to USE$:[MAL]C.DAT;4 (29 records)
```

The APPEND command appends the files A.DAT and B.MEM to the file C.DAT, which must already exist.

4
```
$ APPEND/LOG A.*   B.*
%APPEND-S-APPENDED, USE$:[MAL]A.DAT;5 appended to USE$:[MAL]B.DAT;1 (5 records)
%APPEND-S-APPENDED, USE$:[MAL]A.DOC;2 appended to USE$:[MAL]B.DAT;1 (1 record)
```

Both the input and output file specifications contain wildcard characters in the file type field. The APPEND command appends each file with a file name of A to an existing file with B as its file name. The file type of the first input file located determines the output file type.

5
```
$ APPEND BOSTON"JOHN_SMITH YANKEE"::DEMO01.DAT, DEMO2.DAT
$ _To:   DALLAS::DISK1:[MODEL.TEST]TEST.DAT
```

This APPEND command adds the contents of the files DEMO1.DAT and DEMO2.DAT at remote node BOSTON to the end of the file TEST.DAT at remote node DALLAS.

ASSIGN

Creates a logical name and assigns an equivalence string, or a list of strings, to the specified logical name. If you specify an existing logical name, the new equivalence name replaces the existing equivalence name.

FORMAT

ASSIGN *equivalence-name[,...] logical-name[:]*

PARAMETERS

equivalence-name[,...]

Specifies a character string of 1 to 255 characters. Defines the equivalence name, usually a file specification, device name, or other logical name, to be associated with the logical name in the specified logical name table. If the string contains other than uppercase alphanumeric, dollar sign, or underscore characters, enclose it in quotation marks ("). Use two consecutive quotation marks ("") to denote an actual quotation mark. Specifying more than one equivalence name for a logical name creates a search list.

When you specify an equivalence name that will be used as a file specification, you must include the punctuation marks (colons, brackets, periods) that would be required if the equivalence name were used directly as a file specification. Therefore, if you specify a device name as an equivalence name, terminate the device name with a colon.

The ASSIGN command allows you to assign the same logical name to more than one equivalence name. When you specify more than one equivalence name for a logical name, you create a search list. See Section 4.7 of the *VMS DCL Concepts Manual* for more information on search lists.

logical-name

Specifies the logical name string, which is a character string containing up to 255 characters. You choose a logical name to represent the equivalence name in the specified logical name table.

If the string contains other than uppercase alphanumeric, dollar sign, or underscore characters, enclose it in quotation marks ("). Use two consecutive quotation marks ("") to denote an actual quotation mark. If you terminate the logical-name parameter with a colon, the system removes the colon before placing the name in a logical name table. (This differs from the DEFINE command, which saves the colon.) If the logical name is to be entered into the process directory (LNM$PROCESS_DIRECTORY) or system directory (LNM$SYSTEM_DIRECTORY) logical name tables, then the name may only have from 1 to 31 alphanumeric characters (including the dollar sign and underscore). By default, the logical name is placed in the process logical name table.

If the logical name contains any characters other than alphanumeric characters, the dollar sign, or the underscore, enclose the name in quotation marks. If the logical name contains quotation marks, enclose the name in quotation marks and use two sets of quotation marks in the places where you want one set of quotation marks to occur. Note that if you enclose a name in quotation marks, the case of alphabetic characters is preserved.

DESCRIPTION

The ASSIGN command creates an entry in a logical name table by defining a logical name to stand for one or more equivalence names. An equivalence name can be a device name, another logical name, a file specification, or any other string.

To specify the logical name table where you want to enter a logical name, use the /PROCESS, /JOB, /GROUP, /SYSTEM, or /TABLE qualifier. If you enter more than one of these qualifiers, only the last one entered is accepted. If you do not specify a table, the default is /TABLE=LNM$PROCESS (or /PROCESS).

To specify the access mode of the logical name you are creating, use the /USER_MODE, /SUPERVISOR_MODE, or /EXECUTIVE_MODE qualifiers. If you enter more than one of these qualifiers, only the last one entered is accepted. If you do not specify an access mode, then a supervisor mode name is created. You can create a logical name in the same mode as the table in which you are placing the name or in an outer mode. (User mode is the outermost mode; executive mode is the innermost mode.)

You can enter more than one logical name with the same name in the same logical name table, as long as each name has a different access mode. (However, if an existing logical name within a table has the NO_ALIAS attribute, you cannot use the same name to create a logical name in an outer mode in this table.)

If you create a logical name with the same name, in the same table, and in the same mode as an existing name, the new logical name assignment replaces the existing assignment.

You can also use the DEFINE command to create logical names. To delete a logical name from a table, use the DEASSIGN command.

Note: Avoid assigning a logical name that matches the file name of an executable image in SYS$SYSTEM:. Such an assignment will prohibit you from invoking that image.

For additional information on how to create and use logical names, see Chapter 4 of the *VMS DCL Concepts Manual.*

QUALIFIERS

/EXECUTIVE_MODE

Requires SYSNAM privilege.

Specifies the mode of the logical name. If you specify executive mode, but do not have SYSNAM privilege, the qualifier is ignored and a supervisor mode logical name is created. The mode of the logical name must be the same as or external to (less privileged than) the mode of the table in which you are placing the name.

/GROUP

Requires SYSPRV or GRPNAM privilege.

Places the logical name in the group logical name table. Other users who have the same group number in their user identification codes (UICs) can access the logical name. The /GROUP qualifier is synonymous with /TABLE=LNM$GROUP.

/JOB
Requires SYSPRV or GRPNAM privilege.

Places the logical name in the jobwide logical name table. All processes within the same job tree as the process creating the logical name can access the logical name. The /JOB qualifier is synonymous with /TABLE=LNM$JOB.

/LOG (default)
/NOLOG
Displays a message when a new logical name supersedes an existing name.

/NAME_ATTRIBUTES[=(keyword[,...])]
Specifies the attributes for a logical name. By default, no attributes are set. You can specify the following keywords for attributes:

CONFINE	Does not copy the logical name into a spawned subprocess; relevant only for logical names in a private table.
NO_ALIAS	Prohibits creation of logical names with the same name in an outer (less privileged) access mode within the specified table. If another logical name with the same name and an outer access mode already exists in this table, the name is deleted.

If you specify only one keyword, you can omit the parentheses. Only the attributes you specify are set.

/PROCESS (default)
Places the logical name in the process logical name table. The /PROCESS qualifier is synonymous with /TABLE=LNM$PROCESS.

/SUPERVISOR_MODE (default)
Creates a supervisor mode logical name in the specified table.

/SYSTEM
Requires SYSNAM or SYSPRV privilege.

Places the logical name in the system logical name table. All system users can access the logical name. The /SYSTEM qualifier is synonymous with /TABLE=LNM$SYSTEM.

/TABLE=name
Requires WRITE (W) access to the table if the table is shareable.

Specifies the logical name table in which the logical name is to be entered. You can use the /TABLE qualifier to specify a user-defined logical name table (created with the CREATE/NAME_TABLE command); to specify the process, job, group, or system logical name tables; or to specify the process or system logical name directory tables.

If you specify the table name using a logical name that has more than one translation, the logical name is placed in the first table found. For example, if you specify ASSIGN/TABLE=LNM$FILE_DEV and LNM$FILE_DEV is equated to LNM$PROCESS, LNM$JOB, LNM$GROUP, and LNM$SYSTEM, then the logical name is placed in LNM$PROCESS.

If you do not explicitly specify the /TABLE qualifier, the default is /TABLE=LNM$PROCESS (or /PROCESS).

/TRANSLATION_ATTRIBUTES[=(keyword[,...])]

Equivalence-name qualifier.

Specifies attributes of the equivalence-name parameter. Possible keywords are as follows:

CONCEALED	Indicates that the equivalence string is the name of a concealed device. When a concealed device name is defined, the system displays the logical name, rather than the equivalence string, in messages that refer to the device. If you specified the CONCEALED attribute, then the equivalence string must be a physical device name.
TERMINAL	Indicates that the equivalence string should not be translated iteratively; logical name translation should terminate with the current equivalence string.

If you specify only one keyword, you can omit the parentheses. Only the attributes you specify are set.

Note that different equivalence strings of the same logical name can have different translation attributes specified.

/USER_MODE

Creates a user mode logical name in the specified table.

If you specify a user mode logical name in the process logical name table, that logical name is used for the execution of a single image only; user mode entries are deleted from the logical name table when any image executing in the process exits; that is, after any DCL command or user program that executes an image completes execution.

EXAMPLES

1
```
$ ASSIGN $DISK1:[ACCOUNTS.MEMOS] MEMOSD
```

The ASSIGN command in this example equates the partial file specification $DISK1:[ACCOUNTS.MEMOS] to the logical name MEMOSD.

2
```
$ ASSIGN/USER_MODE $DISK1:[ACCOUNTS.MEMOS]WATER.TXT TM1
```

The ASSIGN command in this example equates the logical name TM1 to a file specification. After the next image runs, the logical name is automatically deassigned.

3
```
$ ASSIGN XXX1:[CHARLES]  CHARLIE
$ PRINT  CHARLIE:TEST.DAT
Job 274 entered on queue SYS$PRINT
```

The ASSIGN command in this example associates the logical name CHARLIE with the directory name [CHARLES] on the disk XXX1. Subsequent references to the logical name CHARLIE result in the correspondence between the logical name CHARLIE and the disk and directory specified. The PRINT command queues a copy of the file XXX1:[CHARLES]TEST.DAT to the system printer.

4
```
$ ASSIGN YYY2:  TEMP:
$ SHOW LOGICAL TEMP
   "TEMP" = "YYY2:" (LNM$PROCESS_TABLE)
$ DEASSIGN TEMP
```

The ASSIGN command in this example equates the logical name TEMP to the device YYY2. TEMP is created in supervisor mode and placed in the process logical name table. The SHOW LOGICAL command verifies that the logical name assignment was made. Note that the logical name TEMP was terminated with a colon in the ASSIGN command, but that the command interpreter deleted the colon before placing the name in the logical name table. Thus, you can specify TEMP without a colon in the subsequent DEASSIGN command. You should omit the colon in the SHOW LOGICAL command (for example, SHOW LOGICAL TEMP).

5
```
$ MOUNT TTT1: MASTER TAPE
$ ASSIGN TAPE:NAMES.DAT PAYROLL
$ RUN PAYROLL
   .
   .
   .
```

The MOUNT command in this example establishes the logical name TAPE for the device TTT1, which has the volume labelled MASTER mounted on it. The ASSIGN command equates the logical name PAYROLL with the file named NAMES.DAT on the logical device TAPE. Thus, an OPEN request in a program referring to the logical name PAYROLL results in the correspondence between the logical name PAYROLL and the file NAMES.DAT on the tape whose volume label is MASTER.

6
```
$ CREATE/NAME_TABLE TABLE1
$ ASSIGN/TABLE=LNM$PROCESS_DIRECTORY TABLE1,-
_$ LNM$PROCESS,LNM$JOB,LNM$GROUP,LNM$SYSTEM LNM$FILE_DEV
$ ASSIGN/TABLE=TABLE1 -
_$ /TRANSLATION_ATTRIBUTES=CONCEALED  DBA1:  WORK_DISK
```

The CREATE/NAME_TABLE command in this example creates the process private logical name table TABLE1.

The first ASSIGN command ensures that TABLE1 is searched first in any logical name translation of a file specification or device name (because TABLE1 is the first item in the equivalence string for the logical name LNM$FILE_DEV, which determines the default search sequence of logical name tables whenever a device or file specification is translated).

The second ASSIGN command assigns the logical name WORK_DISK to the physical device DBA1, and places the name in TABLE1. The logical name has the concealed attribute. Therefore, the logical name WORK_DISK will be displayed in system messages.

7
```
$ ASSIGN/TABLE=LNM$PROCESS/TABLE=LNM$GROUP  DBA0:  SYSFILES
$ SHOW LOGICAL  SYSFILES
  "SYSFILES" = "DBA0:" (LNM$GROUP_000240)
```

The ASSIGN command in this example contains conflicting qualifiers. When you specify conflicting qualifiers, the ASSIGN command uses the last qualifier specified. The response from the SHOW LOGICAL command indicates that the name was placed in the group logical name table.

```
8  $ ASSIGN/TABLE=LNM$GROUP 'F$TRNLNM("SYS$COMMAND")' TERMINAL
   %DCL-I-SUPERSEDE, previous value of TERMINAL has been superseded
```

The ASSIGN command in this example uses the lexical function F$TRNLNM to translate the logical name SYS$COMMAND and use the result as the equivalence name for the logical name TERMINAL. The message from the ASSIGN command indicates that an entry for the logical name TERMINAL already existed in the group logical name table, and that the new entry has replaced the previous one.

If this command is used in a LOGIN.COM file, the entry for TERMINAL will be redefined at the beginning of each terminal session. The current process and any subprocesses it creates can execute images that use the logical name TERMINAL to write messages to the current terminal device.

```
9  $ ASSIGN DALLAS::DMA1:  DATA
```

The ASSIGN command in this example associates the logical name DATA with the device specification DMA1 on remote node DALLAS. Subsequent references to the logical name DATA result in references to the disk on the remote node.

```
10 $ CREATE AVERAGE.COM
   $ ASSIGN/USER_MODE SYS$COMMAND:  SYS$INPUT
   $ EDIT/EDT AVERAGE.FOR
   $ FORTRAN AVERAGE
   $ LINK AVERAGE
   $ RUN AVERAGE
   87
   80
   90
   9999
   $ EXIT
   CTRL/Z
   $ @AVERAGE.COM
```

The CREATE command in this example creates the command procedure AVERAGE.COM. Then the command procedure is executed.

The command procedure uses the ASSIGN command with the /USER_MODE qualifier to change temporarily the value of SYS$INPUT. When the EDT editor is invoked, it accepts input from the terminal. This allows you to create or modify AVERAGE.FOR interactively.

When you exit from EDT, SYS$INPUT is reassigned to its original value (the input stream provided by the command procedure). Thus, when the program AVERAGE.FOR is ready to accept input, it looks for that input in the command procedure.

ASSIGN/MERGE

Removes all jobs from one queue and merges them into another existing queue. Does not affect jobs that are executing.

Requires OPER privilege or EXECUTE access to both queues.

FORMAT

ASSIGN/MERGE *target-queue[:] source-queue[:]*

PARAMETERS

target-queue[:]

Specifies the name of the queue into which the jobs are being merged.

source-queue[:]

Specifies the name of the queue from which the jobs are being removed.

DESCRIPTION

The ASSIGN/MERGE command removes the pending jobs in one queue and places them in another queue. This command does not affect any executing jobs in either the target queue or the source queue. Jobs currently running in the source queue complete in that queue. This command is generally used with printer queues, although it can be used with batch queues.

The ASSIGN/MERGE command is particularly useful when a line printer malfunctions. By entering the ASSIGN/MERGE command, you can reroute existing jobs to a different print device. To perform the merge operation without losing or disrupting any jobs, stop the source queue with the STOP /QUEUE/NEXT command. Then enter the STOP/QUEUE/REQUEUE command to ensure that the current job on the source queue is requeued for processing on the target queue. (If the STOP/QUEUE/REQUEUE command fails to requeue the job, use the STOP/QUEUE/RESET command to regain control of the queue.) Once you enter the STOP commands, enter the ASSIGN/MERGE command.

EXAMPLE

```
$ STOP/QUEUE/NEXT LPB0
$ STOP/QUEUE/REQUEUE=LPA0 LPB0
$ ASSIGN/MERGE LPA0 LPB0
```

In this example, the STOP/QUEUE/NEXT command prevents another job from executing on queue LPB0. The STOP/QUEUE/REQUEUE command requeues the current job running on LPB0 to the target queue LPA0. The ASSIGN/MERGE command removes the remaining jobs from the LPB0 printer queue and places them in the LPA0 printer queue.

ASSIGN/QUEUE

Assigns, or redirects, a logical queue to a single execution queue. ASSIGN /QUEUE can be used only with printer or terminal queues.

Requires OPER privilege or EXECUTE access to both queues.

FORMAT

ASSIGN/QUEUE *queue-name[:] logical-queue-name[:]*

PARAMETERS

queue-name[:]

Name of the execution queue. The queue cannot be a logical queue, a generic queue, or a batch queue.

logical-queue-name[:]

Name of the logical queue.

DESCRIPTION

The ASSIGN/QUEUE command sets up a one-to-one correspondence between a logical queue and an execution queue. Jobs submitted to the logical queue are always queued to the specified execution queue for eventual printing.

When you enter the ASSIGN/QUEUE command, the logical queue cannot be running.

Once you initialize a logical queue, use the ASSIGN/QUEUE command to associate the logical queue with an existing execution queue. You must follow these steps to set up a logical queue:

1. Initialize the logical queue with the INITIALIZE/QUEUE command. (Do not use the /START qualifier.)
2. Assign the logical queue name to an existing execution queue.
3. Start the logical queue with the START/QUEUE command.

After you enter the START/QUEUE command for the logical queue, jobs can be sent to the logical queue for processing.

EXAMPLES

1

```
$ INITIALIZE/QUEUE/DEFAULT=FLAG=ONE/START LPA0
$ INITIALIZE/QUEUE TEST_QUEUE
$ ASSIGN/QUEUE LPA0 TEST_QUEUE
$ START/QUEUE TEST_QUEUE
```

This example first initializes and starts the printer queue LPA0. The LPA0 queue is set to have a flag page precede each job. The second INITIALIZE /QUEUE command creates the logical queue TEST_QUEUE. The ASSIGN /QUEUE command assigns the logical queue TEST_QUEUE to the printer queue LPA0. The START/QUEUE command starts the logical queue.

2 `$ INITIALIZE/QUEUE/START LPB0`

The ASSIGN/QUEUE command is not needed in this example because a logical queue is not being initialized. A printer queue is being initialized; LPB0 is the name of a line printer. After you enter the INITIALIZE/QUEUE /START command, jobs can be queued to LPB0 for printing.

ATTACH

Transfers control from your current process (which then hibernates) to the specified process.

The ATTACH and SPAWN commands cannot be used if your terminal has an associated mailbox.

FORMAT

ATTACH *[process-name]*

PARAMETER

process-name

Specifies the name of a parent process or spawned subprocess to which control passes. The process must already exist, be part of your current job, and share the same input stream as your current process. However, the process cannot be your current process or a subprocess created with the /NOWAIT qualifier.

Process names can contain from 1 to 15 alphanumeric characters. If a connection to the specified process cannot be made, an error message is displayed.

The process-name parameter is incompatible with the /IDENTIFICATION qualifier.

DESCRIPTION

The ATTACH command allows you to connect your input stream to another process. You can use the ATTACH command to change control from one subprocess to another subprocess or to the parent process.

When you enter the ATTACH command, the parent or "source" process is put into hibernation, and your input stream is connected to the specified destination process. You can use the ATTACH command to connect to a subprocess that is part of a current job left hibernating as a result of the SPAWN/WAIT command or another ATTACH command as long as the connection is valid. (No connection can be made to the current process, to a process that is not part of the current job, or to a process that does not exist. If any of these connections are attempted, an error message is displayed.)

You can also use the ATTACH command in conjunction with the SPAWN /WAIT command to return to a parent process without terminating the created subprocess. See the description of the SPAWN command for more details.

QUALIFIER

/IDENTIFICATION=pid

Specifies the process identification (PID) of the process to which terminal control will be transferred. Leading zeros can be omitted. The /IDENTIFICATION qualifier is incompatible with the process-name parameter.

If you omit the /IDENTIFICATION qualifier, you must specify a process name.

ATTACH

EXAMPLES

1 `$ ATTACH JONES_2`

Transfers the terminal's control to the subprocess JONES_2.

2 `$ ATTACH/IDENTIFICATION=30019`

The ATTACH command switches control from the current process to a process having the PID 30019. Notice that because the /IDENTIFICATION qualifier is specified, the process-name parameter is omitted.

BACKUP

Invokes the Backup Utility (BACKUP) to perform one of the following BACKUP operations:

- Make copies of disk files.
- Save disk files as data in a file created by BACKUP on disk or magnetic tape. (Files created by BACKUP are called save sets.)
- Restore disk files from a BACKUP save set.
- Compare disk files or files in a BACKUP save set with other disk files.
- List information about files in a BACKUP save set to an output device or file.

Note that standalone BACKUP cannot be invoked this way, but must be bootstrapped in order to run. For a complete description of the Backup Utility (including information about the BACKUP command and its qualifiers) as well as using standalone BACKUP, see the *VMS Backup Utility Manual.*

FORMAT

BACKUP *input-specifier output-specifier*

CALL

Tranfers control to a labeled subroutine within a command procedure. The CALL command creates a new procedure level as does the @ (execute procedure) command. The SUBROUTINE and ENDSUBROUTINE commands define the beginning and ending of the subroutine. The SUBROUTINE command must be the first executable statement in a subroutine.

FORMAT

CALL *label [p1[p2[... p8]]]*

PARAMETERS

label

Specifies a 1- to 255-alphanumeric character label appearing as the first item on a command line. A label may not contain embedded blanks. When the CALL command is executed, control passes to the command following the specified label.

The label can precede or follow the CALL statement in the current command procedure. A label in a command procedure must be terminated with a colon.

All labels are procedure level dependent except for those labels that define subroutine entry points. The subroutine entry point labels are local to the current command procedure file level and must be unique.

p1 [p2 [... p8]]

Specifies from one to eight optional parameters to pass to the command procedure. Use two consecutive quotation marks ("") to specify a null parameter. The parameters assign character string values to the symbols named P1, P2, and so on in the order of entry, to a maximum of eight. The symbols are local to the specified command procedure. Separate each parameter with one or more blanks.

You can specify a parameter with a character string value containing alphanumeric or special characters, with the following restrictions:

- The command interpreter converts alphabetic characters to uppercase and uses blanks to delimit each parameter. To pass a parameter that contains embedded blanks or lowercase letters, place the parameter in quotation marks.
- If the first parameter begins with a slash character (/), you must enclose the parameter in quotation marks.
- To pass a parameter that contains quotation marks and spaces, enclose the entire string in quotation marks and use two consecutive quotation marks within the string. For example:

  ```
  $ CALL SUB1 "Never say ""quit"""
  ```

  When control transfers to SUB1, the parameter P1 is equated to the string:

  ```
  Never say "quit"
  ```

If a string contains quotation marks and does not contain spaces, the quotation marks are preserved in the string and the letters within the quotation marks remain in lowercase. For example:

```
$ CALL SUB2 abc"def"ghi
```

When control transfers to SUB2, the parameter P1 is equated to the string:

```
ABC"def"GHI
```

To use a symbol as a parameter, enclose the symbol in apostrophes to force symbol substitution. For example:

```
$ NAME = "JOHNSON"
$ CALL INFO 'NAME'
```

The apostrophes cause the value "JOHNSON" to be substituted for the symbol "NAME". Therefore, the parameter "JOHNSON" is passed as P1 to the subroutine INFO.

DESCRIPTION

The CALL command is similar to the @ (Execute Procedure) command in that it creates a new procedure level. The advantage of the CALL command is that it does not require files to be opened and closed to process the procedure. Using the CALL command also makes managing a set of procedures easier because they can all exist in one file rather than in several files.

When you use the CALL command to transfer control to a subroutine, a new procedure level is created and the symbols P1 through P8 are assigned the values of the supplied arguments. Execution then proceeds until an EXIT command is encountered. At this point, control is transferred to the command line following the CALL command.

Procedures can be nested to a maximum of 32 levels. This includes any combination of command procedure and subroutine calls. Local symbols and labels defined within a nested subroutine structure are treated the same way as if the routines had been invoked with the @ command: that is, labels are only valid for the subroutine level in which they are defined. Local symbols defined in an outer subroutine level are available to any subroutine levels at an inner nesting level.

The SUBROUTINE and ENDSUBROUTINE commands define the beginning and end of a subroutine. The label defining the entry point to the subroutine must appear either immediately before the SUBROUTINE command or on the same command line.

A subroutine can have only one entry point. The subroutine must begin with the SUBROUTINE command as the first executable statement. If an EXIT command is not specified in the procedure, the ENDSUBROUTINE command functions as an EXIT command.

The SUBROUTINE command performs two different functions depending on the context in which it is executed. If executed as the result of a CALL command, it initiates a new procedure level, defines the P1-P8 parameters as specified in the CALL statement, and begins execution of the subroutine. If the SUBROUTINE verb is encountered in the execution flow of the procedure without having been invoked by a CALL command, all the commands following the SUBROUTINE command are skipped until the corresponding ENDSUBROUTINE command is encountered. Although commands are

skipped, all subroutine entry point labels will be defined in the symbol table.

Note: The SUBROUTINE and ENDSUBROUTINE commands cannot be abbreviated to fewer than four characters.

QUALIFIER

/OUTPUT=file-spec

Writes all output to the file or device specified. By default, the output is written to the current SYS$OUTPUT device and the output file type is LIS. System responses and error messages are written to SYS$COMMAND as well as to the specified file. If you specify /OUTPUT, the qualifier must immediately follow the CALL command. No wildcard characters are allowed in the output file specification.

You can also redefine SYS$OUTPUT to redirect the output from a command procedure. If you place the following command as the first line in a command procedure, output will be directed to the file you specify:

```
$ DEFINE SYS$OUTPUT file-spec
```

When the procedure exits, SYS$OUTPUT is restored to its original equivalence string. This produces the same result as using the /OUTPUT qualifier when you execute the command procedure.

EXAMPLE

```
$
$! CALL.COM
$
$! Define subroutine SUB1
$!
$ SUB1: SUBROUTINE
   .
   .
   .
$ CALL SUB2 !Invoke SUB2 from within SUB1
   .
   .
   .
$ @FILE  !Invoke another procedure command file
   .
   .
   .
$ EXIT
$ ENDSUBROUTINE !End of SUB1 definition
$!
$! Define subroutine SUB2
$!
$ SUB2: SUBROUTINE
   .
   .
   .
$ EXIT
$ ENDSUBROUTINE !End of SUB2 definition
$!
$! Start of main routine. At this point, both SUB1 and SUB2
$! have been defined but none of the previous commands have
$! been executed.
```

```
$!
$ START:
$ CALL/OUTPUT=NAMES.LOG SUB1 "THIS IS P1"
   .
   .
   .
$ CALL SUB2 "THIS IS P1" "THIS IS P2"
   .
   .
   .
$ EXIT  !Exit this command procedure file
```

The command procedure in this example shows how to use CALL to transfer control to labeled subroutines. The example also shows that you can call a subroutine or another command file from within a subroutine. The CALL command invokes the subroutine SUB1, directing output to the file NAMES.LOG and allowing other users write access to the file. The subroutine SUB2 is called from within SUB1. The procedure executes SUB2 and then uses the @ (Execute Procedure) command to invoke the command procedure FILE.COM. When all the commands in SUB1 have executed, the CALL command in the main procedure calls SUB2 a second time. The procedure continues until SUB2 has executed.

CANCEL

Cancels wakeup requests for a specified process, including wakeups scheduled with either the RUN command or the $SCHDWK system service.

Requires one of the following:

- **Ownership of the process.**
- **GROUP privilege to cancel scheduled wakeups for processes in the same group but not owned by you.**
- **WORLD privilege to cancel scheduled wakeups for any process in the system.**

FORMAT

CANCEL *[process-name]*

PARAMETER

process-name

Specifies a string of 1 to 15 alphanumeric characters. Specifies the name of the process for which wakeup requests are to be canceled. The specified process must have the same group number in its user identification code (UIC) as the current process. If both the /IDENTIFICATION qualifier and the process name are specified, the process name is ignored. If neither the process-name parameter nor the /IDENTIFICATION qualifier are specified, the CANCEL command cancels scheduled wakeup requests for the current (that is, the issuing) process.

DESCRIPTION

The CANCEL command cancels scheduled wakeup requests for the specified process.

The CANCEL command does not delete the specified process. If the process is executing an image when the CANCEL command is issued for it, the process hibernates instead of exiting after the image completes execution.

To delete a hibernating process for which wakeup requests have been canceled, use the STOP command. You can determine whether a subprocess has been deleted by entering the SHOW PROCESS command with the /SUBPROCESSES qualifier.

QUALIFIER

/IDENTIFICATION=pid

Identifies the process by its process identification (PID). You can omit leading zeros when you specify the PID.

EXAMPLES

1
```
$ CANCEL CALENDAR
```

The CANCEL command in this example cancels a wakeup request for a process named CALENDAR (which continues to hibernate until deleted with the STOP command).

2
```
$ RUN/SCHEDULE=14:00 STATUS
%RUN-S-PROC_ID, identification of created process is 0013012A
    .
    .
    .
$ CANCEL/IDENTIFICATION=13012A
```

The RUN command in this example creates a process to execute the image STATUS. The process hibernates and is scheduled to be awakened at 14:00. Before the process is awakened, the CANCEL command cancels the wake-up request.

3
```
$ RUN/PROCESS_NAME=LIBRA/INTERVAL=1:00    LIBRA
%RUN-S-PROC_ID, identification of created process is 00130027
    .
    .
    .
$ CANCEL LIBRA
$ STOP LIBRA
```

The RUN command in this example creates a subprocess named LIBRA to execute the image LIBRA.EXE at hourly intervals.

Subsequently, the CANCEL command cancels the wakeup request. The process continues to exist, but in a state of hibernation, until the STOP command deletes it.

CLOSE

Closes a file opened with the OPEN command and deassigns the associated logical name.

FORMAT

CLOSE *logical-name[:]*

PARAMETER

logical-name[:]

Specifies the logical name assigned to the file when it was opened with the OPEN command.

DESCRIPTION

Files that are opened for reading or writing at the command level remain open until closed with the CLOSE command, or until the process terminates. If a command procedure that opens a file terminates without closing the open file, the file remains open; the command interpreter does not automatically close it.

QUALIFIERS

/ERROR=label

Specifies a label in the command procedure to receive control if the CLOSE operation results in an error. Overrides any ON condition action specified. If an error occurs and the target label is successfully given control, the global symbol $STATUS retains the code for the error that caused the error path to be taken.

/LOG (default)
/NOLOG

Generates a warning message when you attempt to close a file that was not opened by DCL. If you specify the /ERROR qualifier, the /LOG qualifier has no effect. If the file has not been opened by DCL, the error branch is taken and no message is displayed.

EXAMPLES

1
```
$ OPEN/READ INPUT_FILE   TEST.DAT
$ READ_LOOP:
$ READ/END_OF_FILE=NO_MORE  INPUT_FILE  DATA_LINE
    .
    .
    .
$ GOTO READ_LOOP
$ NO_MORE:
$ CLOSE INPUT_FILE
```

The OPEN command in this example opens the file TEST.DAT and assigns it the logical name of INPUT_FILE. The /END_OF_FILE qualifier on the READ command requests that, when the end-of-file is reached, the command interpreter should transfer control to the line at the label NO_MORE. The CLOSE command closes the input file.

```
2  $ @READFILE
   CTRL/Y
   $ STOP
   $ SHOW LOGICAL/PROCESS
       .
       .
       .
     "INFILE" = "_DB1"
     "OUTFILE" = "_DB1"
   $ CLOSE INFILE
   $ CLOSE OUTFILE
```

In this example, CTRL/Y interrupts the execution of the command procedure READFILE.COM. Then, the STOP command stops the procedure. The SHOW LOGICAL/PROCESS command displays the names that currently exist in the process logical name table. Among the names listed are the logical names INFILE and OUTFILE, assigned by OPEN commands in the procedure READFILE.COM.

The CLOSE commands close these files and deassign the logical names.

CONNECT

Connects your physical terminal to a virtual terminal that is connected to another process.

You must connect to a virtual terminal that is connected to a process with your user identification code (UIC). No other physical terminals may be connected to the virtual terminal.

FORMAT

CONNECT *virtual-terminal-name*

PARAMETER

virtual-terminal-name

Specifies the name of the virtual terminal to which you are connecting. A virtual terminal name always begins with VTA. To determine the name of the virtual terminal that is connected to a process, enter the SHOW USERS command.

DESCRIPTION

The CONNECT command connects you to a separate process, as opposed to the SPAWN and ATTACH commands, which create and attach subprocesses.

The CONNECT command is useful when you are logged in to the system using telecommunications lines. If there is noise over the line and you lose the carrier signal, your process does not terminate. After you log in again, you can reconnect to the original process and log out of your second process.

To use the CONNECT command, the virtual terminal feature must be enabled for your system with the System Generation Utility (SYSGEN). If virtual terminals are allowed on your system, then the SET TERMINAL /PERMANENT command is used to enable the virtual terminal characteristic for a particular physical terminal. When this characteristic is enabled, a virtual terminal will be created when a user logs in on the physical terminal. The physical terminal is connected to the virtual terminal which is, in turn, connected to the process.

When the connection between the physical terminal and the virtual terminal is broken, the process remains connected to the virtual terminal. If the process is executing an image, it continues until it needs terminal input or attempts to write to the terminal. At that point, it waits.

You can connect to a virtual terminal even if you are not currently using a virtual terminal. However, you must use the CONNECT command with the /LOGOUT qualifier to log out of your current process. If you connect to a virtual terminal from another virtual terminal, you can save your current process by using the /NOLOGOUT qualifier.

QUALIFIERS

/CONTINUE
/NOCONTINUE (default)

Controls whether the CONTINUE command is executed in the current process just before connecting to another process. This allows an interrupted image to continue processing after you connect to another process.

The /CONTINUE qualifier is incompatible with the /LOGOUT qualifier.

/LOGOUT (default)
/NOLOGOUT

Logs out your current process when you connect to another process using a virtual terminal.

When you enter the CONNECT command from a process that is not connected to a virtual terminal, you must specify the /LOGOUT qualifier. Otherwise, DCL will issue an error message.

The /LOGOUT qualifier is incompatible with the /CONTINUE qualifier.

EXAMPLES

1

```
$ RUN AVERAGE
CTRL/Y
$ CONNECT/CONTINUE VTA72
```

In this example, the RUN command is used to execute the image AVERAGE.EXE. This command is entered from a terminal that is connected to a virtual terminal. Next, CTRL/Y is entered to interrupt the image. After you interrupt the image, enter the CONNECT command with the /CONTINUE qualifier. This issues the CONTINUE command, so the image continues to run and connects you to another virtual terminal. You can reconnect to the process later.

2

```
$ SHOW USERS
          VMS Interactive Users
          23-JUN-1988 15:25:30.75
    Total number of interactive users = 5
 Username     Process Name      PID     Terminal
 REICH        REICH           2040055C  VTA267:        TXC13:
 GLASS        Phil            20400560  VTA270:        LTA102:
 ADAMS        ADAMS           20400551  VTA261:        TTC7:
 JANZEN       JANZEN          2040056D  VTA272:        Disconnected
 JANZEN       _VTA273:        2040056E  VTA273:        TTB5:
$ CONNECT VTA273
  JANZEN        logged out at 23-JUN-1988 15:26:56.53
$
```

This example shows how to reconnect to your original process after you have lost the carrier signal. First, you must log in again and create a new process. After you log in, enter the SHOW USERS command to determine the virtual terminal name for your initial process. Then enter the CONNECT command to connect to the virtual terminal associated with your original process. The process from which you enter the CONNECT command is logged out because no qualifiers are specified.

When reconnecting to the original process, the user continues running the image you were running when you lost the carrier signal. In this example, the user JANZEN was at DCL level when the connection was broken.

CONTINUE

Resumes execution of a DCL command, a program, or a command procedure that was interrupted by CTRL/Y or CTRL/C. You cannot resume execution of the image if you have entered a command that executes another image or if you have invoked a command procedure. You can abbreviate the CONTINUE command to a single letter, C.

The CONTINUE command serves as the target command of an IF or ON command in a command procedure. The CONTINUE command is also a target command when it follows a label that is the target of a GOTO command. In addition, you can use the CONTINUE command to resume processing of a program that has executed either a VAX FORTRAN PAUSE statement or a VAX COBOL-74 STOP literal statement.

FORMAT

CONTINUE

PARAMETERS

None.

DESCRIPTION

The CONTINUE command enables you to resume processing an image that was interrupted by CTRL/Y or CTRL/C. If the interruption involved execution of another image, you cannot use the CONTINUE command on the original image. Similarly, if you have invoked a command procedure after interrupting the original image, that image cannot resume processing. However, you can use CONTINUE after commands that do not execute separate images; see Table 1–1 in the *VMS DCL Concepts Manual* for a list of these commands.

EXAMPLES

1

```
$ RUN  MYPROGRAM_A
CTRL/Y
$ SHOW TIME
  15-APR-1988 13:40:12
$ CONTINUE
```

In this example, the RUN command executes the program MYPROGRAM_A. While the program is running, pressing CTRL/Y interrupts the image. The SHOW TIME command requests a display of the current date and time. The CONTINUE command resumes the image.

2

```
$ ON SEVERE_ERROR THEN CONTINUE
```

In this example, the command procedure statement requests the command interpreter to continue executing the procedure if any warning, error, or severe error status value is returned from the execution of a command or program. This ON statement overrides the default action, which is to exit from a procedure following errors or severe errors.

CONVERT

Invokes the Convert Utility (CONVERT) to copy records from one file to another, changing the organization and format of the input file to those of the output file. For a complete description of the Convert Utility, including more information about the CONVERT command and its qualifiers, see the *VMS Convert and Convert/Reclaim Utility Manual.*

FORMAT

CONVERT *input-file-spec[,...] output-file-spec*

CONVERT/RECLAIM

Invokes the Convert/Reclaim Utility (CONVERT/RECLAIM) to make empty buckets in Prolog 3 indexed files available so that new records can be written in them. If all the records in a bucket have been deleted, that bucket is locked until CONVERT/RECLAIM makes it available. Unlike CONVERT, CONVERT/RECLAIM maintains record file addresses (RFAs). The /RECLAIM qualifier is required. For a complete description of the Convert/Reclaim Utility, including more information about the CONVERT /RECLAIM command and its qualifier, see the Convert Utility in the *VMS Convert and Convert/Reclaim Utility Manual*.

FORMAT CONVERT/RECLAIM *file-spec*

COPY

Creates a new file from one or more existing files. If device or directory is not specified, your current default device and directory are used. The COPY command can do the following:

- Copy an input file to an output file
- Concatenate two or more input files into a single output file
- Copy a group of input files to a group of output files

FORMAT

COPY *input-file-spec[,...] output-file-spec*

PARAMETERS

input-file-spec[,...]

Specifies the name of an existing file to be copied. Wildcard characters are allowed. Use a plus sign (+) or a comma (,) to indicate multiple file specifications.

output-file-spec

Specifies the name of the output file into which the input is copied.

You must specify at least one field in the output file specification. If the device or directory is not specified, your current default device and directory are used. The COPY command replaces any other missing fields (file name, file type, version number) with the corresponding field of the input file specification. If you specify more than one input file, the COPY command generally uses the fields from the first input file to determine any missing fields in the output file.

The asterisk wildcard character can be used in place of any two of the following: the file name, file type, or version number. The COPY command uses the corresponding field in the related input file to name the output file. The wildcard character can also be used in the output file specification to have COPY create more than one output file. For example:

```
$ COPY A.A;1, B.B;1 *.C
```

This COPY command creates the files A.C;1 and B.C;1 in the current default directory.

DESCRIPTION

The COPY command, by default, creates a single output file. When more than one input file is specified, the first input file is copied to the output file, and each subsequent input file is appended to the end of the output file. If a field of the output file specification is missing or contains an asterisk wildcard character, the COPY command uses the corresponding field from the first, or only, input file to name the output file.

If multiple input files with maximum record lengths are specified, the output file is given the maximum record length of the first input file. If the COPY command encounters a record in a subsequent input file that is longer than the maximum record length of the output file, it issues a message noting the incompatible file attributes and begins copying the next file.

To create multiple output files, specify multiple input files and use at least one of the following:

- An asterisk wildcard character in the output directory specification, file name, file type, or version number field
- Only a node name, a device name, or a directory specification as the output file specification
- The /NOCONCATENATE qualifier

When multiple output files are created, the corresponding field from each input file is used in the output file name.

You can use the /LOG qualifier when you specify multiple input and output files to verify that the files were copied as you intended.

Version Numbers

If no version numbers are specified for input and output files, the COPY command (by default) assigns a version number to the output files that is either of the following:

- The version number of the input file
- A version number one greater than the highest version number of an existing file with the same file name and file type

When the output file version number is specified by an asterisk wildcard character, the COPY command uses the version numbers of the associated input files as the version numbers of the output files.

If the output file specification has an explicit version number, the COPY command uses that number for the output file specification. If a higher version of the output file already exists, a warning message is issued, but the file is still copied. If an equal version of the output file already exists, a message is issued and the input file is *not* copied.

File Protection and Creation/Revision Dates

The COPY command considers an output file to be new when any portion of the output file name is explicitly specified. The creation date for a new file is set to the current time and date.

If the output file specification has one or more wildcard characters, the creation date of the input file is used.

The revision date of the output file is always set to the current time and date; the backup date is set to zero. The output file is assigned a new expiration date. (Expiration dates are set by the file system if retention is enabled; otherwise they are set to zero.)

The protection and access control list (ACL) of the output file is determined by the following parameters, in the following order:

- Protection of previously existing versions of the output file
- Default Protection and ACL of the output directory
- Process default file protection.

(Note that the BACKUP command takes the creation and revision dates as well as the file protection from the input file.)

Use the /PROTECTION qualifier to change the output file protection.

Normally, the owner of the output file will be the same as the creator of the output file. However, if a user with extended privileges creates the output file, the owner will be the owner of the parent directory or a previous version of the output file if it exists.

Extended privileges include any of the following:

- SYSPRV or BYPASS
- System UIC
- GRPPRV if the owner of the parent directory (or previous version of the output file) is in the same group as the creator of the new output file
- An identifier (with the resource attribute) representing the owner of the parent directory (or previous version of the output file)

Copying Directory Files

If you copy a file that is a directory, a new empty directory is created as a subdirectory of the named directory. Note that even if the input directory contained files, none of those files are copied to the new subdirectory. For example:

```
$ COPY  [SMITH]CATS.DIR  [JONES]
```

This COPY command creates the new subdirectory [JONES]CATS.DIR, which is empty. The files contained in the directory [SMITH]CATS.DIR can be copied once the new subdirectory [JONES]CATS.DIR is created.

QUALIFIERS

/ALLOCATION=n

Output-file-spec qualifier.

Forces the initial allocation of the output file to the number of 512-byte blocks specified by n. If not specified, the initial allocation of the output file is determined by the size of the input file being copied.

/BACKUP

Modifies the time value specified with the /BEFORE or /SINCE qualifier. /BACKUP selects files according to the dates of their most recent backups. This qualifier is incompatible with the other qualifiers that also allow you to select files according to time attributes: /CREATED, /EXPIRED, and /MODIFIED. If you specify none of these four time qualifiers, the default is /CREATED.

/BEFORE[=time]

Selects only those files dated prior to the specified time. You can specify time as an absolute time, as a combination of absolute and delta times, or as one of the following keywords: TODAY (default), TOMORROW, or YESTERDAY. Specify one of the following qualifiers with /BEFORE to indicate the time attribute to be used as the basis for selection: /BACKUP, /CREATED (default), /EXPIRED, or /MODIFIED.

See Section 1.4 of the *VMS DCL Concepts Manual* for complete information on specifying time values.

/BY_OWNER[=uic]

Selects only those files whose owner user identification code (UIC) matches the specified owner UIC. The default UIC is that of the current process.

Specify the UIC using standard UIC format as described in Section 8.1 of the *VMS DCL Concepts Manual*.

/CONCATENATE (default)
/NOCONCATENATE

Creates one output file from multiple input files when wildcard characters are not used in the output file specification. A specification of /NOCONCATENATE generates multiple output files. A wildcard character in an input file specification results in a single output file consisting of the concatenation of all input files matching the file specification.

Files from Files–11 Structure Level 2 disks are concatenated in alphanumeric order; if you specify a wildcard in the file version field, files are copied in descending order by version number. Files from Files–11 Structure Level 1 disks are concatenated in random order.

/CONFIRM
/NOCONFIRM (default)

Controls whether a request is issued before each COPY operation to confirm that the operation should be performed on that file. The following responses are valid:

YES	NO	QUIT
TRUE	FALSE	CTRL/Z
1	0	ALL
	RET	

You can use any combination of uppercase and lowercase letters for word responses. Word responses can be abbreviated to one or more letters (for example, T, TR, or TRU for TRUE), but these abbreviations must be unique. Affirmative answers are YES, TRUE, and 1. Negative answers are NO, FALSE, 0, and the RETURN key. QUIT or CTRL/Z indicates that you want to stop processing the command at that point. When you respond with ALL, the command continues to process but no further prompts are given. If you type a response other than one of those in the list, DCL issues an error message and redisplays the prompt.

/CONTIGUOUS
/NOCONTIGUOUS

Output-file-spec qualifier.

Specifies that the output file must occupy contiguous physical disk blocks. By default, the COPY command creates an output file in the same format as the corresponding input file and does not report an error if not enough space exists for a contiguous allocation. If you copy multiple input files of different formats, the output file may or may not be contiguous. You can use the /CONTIGUOUS qualifier to ensure that files are copied contiguously.

The /CONTIGUOUS qualifier has no effect when you copy files to or from tapes because the size of the file on tape cannot be determined until after it is copied to the disk. If you copy a file from a tape and want the file to be

contiguous, use the COPY command twice: once to copy the file from the tape, and a second time to create a contiguous file.

/CREATED (default)

Modifies the time value specified with the /BEFORE or /SINCE qualifier. The /CREATED qualifier selects files based on their dates of creation. This qualifier is incompatible with the other qualifiers that also allow you to select files according to time attributes: /BACKUP, /EXPIRED, and /MODIFIED. If you specify none of these four time qualifiers, the default is /CREATED.

/EXCLUDE=(file-spec[,...])

Excludes the specified files from the COPY operation. You can include a directory but not a device in the file specification. Wildcard characters are allowed in the file specification. However, you cannot use relative version numbers to exclude a specific version. If you provide only one file specification, you can omit the parentheses.

/EXPIRED

Modifies the time value specified with the /BEFORE or /SINCE qualifiers. /EXPIRED selects files according to their expiration dates. (The expiration date is set with the SET FILE/EXPIRATION_DATE command.) The /EXPIRED qualifier is incompatible with the other qualifiers that also allow you to select files according to time attributes: /BACKUP, /CREATED, and /MODIFIED. If you specify none of these four time qualifiers, the default is /CREATED.

/EXTENSION=n

Output-file-spec qualifier.

Specifies the number of blocks to be added to the output file each time the file is extended. If you do not specify /EXTENSION, the default extension attribute of the output file is determined by the extension attribute of the corresponding input file.

/LOG
/NOLOG (default)

Controls whether the COPY command displays the file specifications of each file copied.

When you use the /LOG qualifier, the COPY command displays the following for each copy operation: (1) the file specifications of the input and output files, (2) the number of blocks or the number of records copied (depending on whether the file is copied on a block-by-block or record-by-record basis), and (3) the total number of new files created.

/MODIFIED

Modifies the time value specified with the /BEFORE or /SINCE qualifier. The /MODIFIED qualifier selects files according to the dates on which they were last modified. This qualifier is incompatible with the other qualifiers that also allow you to select files according to time attributes: /BACKUP, /CREATED, and /EXPIRED. If you specify none of these four time modifiers, the default is /CREATED.

/OVERLAY
/NOOVERLAY (default)
Output-file-spec qualifier.

Requests that data in the input file be copied into the existing specified file, overlaying the existing data, rather than allocating new space for the file. The physical location of the file on disk does not change.

The /OVERLAY qualifier is ignored if the output file is written to a non-file-structured device.

/PROTECTION=(code)
Output-file-spec qualifier.

Specifies protection for the output file. Specify ownership as SYSTEM, OWNER, GROUP, or WORLD and access as R (read), W (write), E (execute), or D (delete). The default protection is that of the existing output file. If no output file exists, the current default protection applies.

See Section 8.1 of the *VMS DCL Concepts Manual* for more information on specifying protection code.

/READ_CHECK
/NOREAD_CHECK (default)
Input-file-spec qualifier.

Reads each record in the input files twice to verify that it has been read correctly.

/REPLACE
/NOREPLACE (default)
Output-file-spec qualifier.

Requests that, if a file already exists with the same file specification as that entered for the output file, the existing file is to be deleted. The COPY command allocates new space for the output file. In general, when you use the /REPLACE qualifier, you will want to include version numbers with the file specifications. By default, the COPY command creates a new version of a file if a file with that specification already exists, incrementing the version number. With /NOREPLACE, an error is signaled when a conflict in version numbers occurs.

/SINCE[=time]
Selects only those files dated after the specified time. You can specify time as an absolute time, a combination of absolute and delta times, or as one of the following keywords: TODAY (default), TOMORROW, or YESTERDAY. Specify one of the following qualifiers with /BEFORE to indicate the time attribute to be used as the basis for selection: /BACKUP, /CREATED (default), /EXPIRED, or /MODIFIED.

See Section 1.4 of the *VMS DCL Concepts Manual* for complete information on specifying time values.

/TRUNCATE
/NOTRUNCATE (default)

Output-file-spec qualifier.

Controls whether or not the COPY command truncates an output file at the end-of-file when copying it. By default, the size of the output file is determined by the allocation of the input file.

/VOLUME=n

Output-file-spec qualifier.

Places the output file on the specified relative volume number of a multivolume set. By default, the output file is placed arbitrarily in a multivolume set.

/WRITE_CHECK
/NOWRITE_CHECK (default)

Output-file-spec qualifier.

Reads each record in the output file after it was written to verify that the record was successfully copied and that the file can subsequently be read without error.

EXAMPLES

1

```
$ COPY TEST.DAT NEWTEST.DAT
```

In this example, the COPY command copies the contents of the file TEST.DAT from the default disk and directory to a file named NEWTEST.DAT on the same disk and directory. If a file named NEWTEST.DAT already exists, the COPY command creates a new version of it.

2

```
$ COPY ALPHA.TXT TMP
$ COPY ALPHA.TXT .TMP
```

In this example, the first COPY command copies the file ALPHA.TXT into a file named TMP.TXT. The COPY command uses the file type of the input file to complete the file specification for the output file. The second COPY command creates a file named ALPHA.TMP. The COPY command uses the file name of the input file to name the output file.

3

```
$ COPY/LOG TEST.DAT  NEW.DAT;1/REPLACE
%COPY-I-REPLACED, DBA0:[MAL]NEW.DAT;1 being replaced
%COPY-S-COPIED, DBA0:[MAL]TEST.DAT;1 copied to DBA0:[MAL]NEW.DAT;1 (1 block)
```

In this example, the /REPLACE qualifier requests that the COPY command replace an existing version of the output file with the new file. The first message from the COPY command indicates that it is replacing an existing file. The version number in the output file must be explicit; otherwise, the COPY command creates a new version of the file NEW.DAT.

4 `$ COPY *.COM [MALCOLM.TESTFILES]`

In this example, the COPY command copies the highest versions of files in the current default directory with the file type COM to the subdirectory MALCOLM.TESTFILES.

5
```
$ COPY/LOG *.TXT *.OLD
%COPY-S-COPIED, DBA0:[MAL]A.TXT;2 copied to DBA0:[MAL]A.OLD;2 (1 block)
%COPY-S-COPIED, DBA0:[MAL]B.TXT;2 copied to DBA0:[MAL]B.OLD;2 (1 block)
%COPY-S-COPIED, DBA0:[MAL]G.TXT;2 copied to DBA0:[MAL]G.OLD;2 (4 blocks)
%COPY-S-NEWFILES, 3 files created
```

In this example, the COPY command copies the highest versions of files with file types of TXT into new files. Each new file has the same file name as an existing file, but a file type of OLD. The last message from the COPY command indicates the number of new files that have been created.

6
```
$ COPY/LOG  A.DAT,B.MEM C.*
%COPY-S-COPIED, DBA0:[MAL]A.DAT;5 copied to DBA0:[MAL]C.DAT;11 (1 block)
%COPY-S-COPIED, DBA0:[MAL]B.MEM;2 copied to DBA0:[MAL]C.MEM;24 (58 records)
%COPY-S-NEWFILES, 2 files created
```

In this example, the two input file specifications are separated with a comma. The asterisk wildcard character in the output file specification indicates that two output files are to be created. For each copy operation, the COPY command uses the file type of the input file to name the output file.

7
```
$ COPY/LOG *.TXT TXT.SAV
%COPY-S-COPIED, DBA0:[MAL]A.TXT;2 copied to DBA0:[MAL]TXT.SAV;1 (1 block)
%COPY-S-APPENDED, DBA0:[MAL]B.TXT;2 appended to DBA0:[MAL]TXT.SAV;1 (3 records)
%COPY-S-APPENDED, DBA0:[MAL]G.TXT;2 appended to DBA0:[MAL]TXT.SAV;1 (51 records)
%COPY-S-NEWFILES, 1 file created
```

In this example, the COPY command copies the highest versions of all files with the file type TXT to a single output file named TXT.SAV. After the first input file is copied, the messages from the COPY command indicate that subsequent files are being appended to the output file.

Note that, if you use the /NOCONCATENATE qualifier in this example, the COPY command creates as many TXT.SAV files as there are input files. Each TXT.SAV file has a different version number.

8 `$ COPY MASTER.DOC DBA1:[BACKUP]`

In this example, the COPY command copies the highest version of the file MASTER.DOC to the device DBA1. If no file named MASTER.DOC already exists in the directory [BACKUP], the COPY command assigns the version number of the input file to the output file. You must have W (write) access to the directory [BACKUP] on device DBA1 for the command to work.

9 `$ COPY SAMPLE.EXE DALLAS::DISK2:[000,000]SAMPLE.EXE/CONTIGUOUS`

In this example, the COPY command copies the file SAMPLE.EXE on the local node to a file with the same name at remote node DALLAS. The /CONTIGUOUS qualifier indicates that the output file is to occupy consecutive physical disk blocks. You must have W (write) access to the device DISK2 on remote node DALLAS for the command to work.

10 `$ COPY *.* PRTLND::*.*`

In this example, the COPY command copies all files within the user directory at the local node to the remote node PRTLND. The new files have the same names as the input file. You must have W (write) access to the default directory on remote node PRTLND for the command to work.

11
```
$ COPY BOSTON::DISK2:TEST.DAT;5
_To: DALLAS"SAM SECRET"::DISK0:[MODEL.TEST]TEST.DAT/ALLOCATION=50
```

In this example, the COPY command copies the file TEST.DAT;5 on the device DISK2 at node BOSTON to a new file named TEST.DAT at remote node DALLAS. The /ALLOCATE qualifier initially allocates 50 blocks for the new file TEST.DAT at node DALLAS. The access control string SAM SECRET is used to access the remote directory.

12
```
$ MOUNT   TAPED1:   VOL025   TAPE:
$ COPY  TAPE:*.*   *
```

In this example, the MOUNT command requests that the volume labeled VOL025 be mounted on the magnetic tape device TAPED1 and assigns the logical name TAPE to the device.

The COPY command uses the logical name TAPE as the input file specification, requesting that all files on the magnetic tape be copied to the current default disk and directory. All the files copied retain their file names and file types.

13
```
$ ALLOCATE CR:
  _CR1: ALLOCATED
$ COPY CR1: CARDS.DAT
$ DEALLOCATE CR1:
```

In this example, the ALLOCATE command allocates a card reader for exclusive use by the process. The response from the ALLOCATE command indicates the device name of the card reader, CR1.

After the card reader is allocated, you can place a deck of cards in the reader and enter the COPY command specifying the card reader as the input file. The COPY command reads the cards into the file CARDS.DAT. The end-of-file in the card deck must be indicated with an EOF card (12-11-0-1-6-7-8-9 overpunch).

The DEALLOCATE command relinquishes use of the card reader.

CREATE

Creates a sequential text file (or files). Specify the content of the file on the lines following the command, one record per line. In interactive mode, terminate the file input with CTRL/Z. In a command procedure, terminate the file input with a line beginning with a dollar sign in column 1 (or with the end of the command procedure).

FORMAT

CREATE *file-spec[,...]*

PARAMETER

file-spec[,...]

Specifies the name of one or more input files to be created. Wildcard characters are not allowed. If you omit either the file name or the file type, the CREATE command does not supply any defaults. The file name or file type is null. If the specified file already exists, a new version is created.

DESCRIPTION

The CREATE command creates a new sequential disk file. The contents of the file are determined by what you enter after the command line. Each separate line that you enter becomes a record in the newly created file. When you have finished entering the records, press CTRL/Z to signal the end of the input.

When you enter the CREATE command from a command procedure file, the system reads all subsequent records in the command procedure file into the new file until it encounters a dollar sign in the first position in a record.

If you use an existing file specification with the CREATE command, the newly created file has a higher version number than any existing files with the same specification.

If you use the CREATE command to create a file in a logical name search list, the file will only be created in the first directory produced by the logical name translation.

Normally, the owner of the output file will be the same as the creator of the output file. However, if a user with extended privileges creates the output file, the owner will be the owner of the parent directory or any previous versions of the output file.

Extended privileges include any of the following:

- SYSPRV or BYPASS
- System UIC
- GRPPRV if the owner of the parent directory (or previous version of the output file) is in the same group as the creator of the new output file
- An identifier (with the resource attribute) representing the owner of the parent directory (or previous version of the output file)

QUALIFIERS

/LOG
/NOLOG (default)

Displays the file specification of each new file created as the command executes.

/OWNER_UIC=uic

Requires SYSPRV privilege to specify a UIC other than your own.

Specifies the user identification code (UIC) to be associated with the file being created. Specify the UIC using standard UIC format as described in Section 8.1 of the *VMS DCL Concepts Manual*.

/PROTECTION=(code)

Specifies protection for the file. Specify ownership as SYSTEM, OWNER, GROUP, or WORLD and protection as R (read), W (write), E (execute), or D (delete). If you do not specify a value for each access category, or if you omit the /PROTECTION qualifier, the command applies the current default protection for each unspecified category.

See Section 8.1 of the *VMS DCL Concepts Manual* for more information on specifying protection code.

The command applies the protection of the existing file to the new file under the following conditions:

- If you specify an existing file specification, and do not specify a value for each access category
- If you omit the /PROTECTION qualifier

/VOLUME=n

Places the file on the specified relative volume of a multivolume set. By default, the file is placed arbitrarily in a multivolume set.

EXAMPLES

1
```
$ CREATE MEET.TXT
John, Residents in the apartment complex will hold their annual meeting
this evening.  We hope to see you there, Regards, Elwood
CTRL/Z
```

The CREATE command in this example creates a text file named MEET.TXT in your default directory. The text file MEET.TXT contains the lines that follow until the CTRL/Z.

CREATE

2

```
$ CREATE A.DAT, B.DAT
Input line one for A.DAT...
Input line two for A.DAT...
   .
   .
   .
CTRL/Z
Input line one for B.DAT...
Input line two for B.DAT...
   .
   .
   .
CTRL/Z
$
```

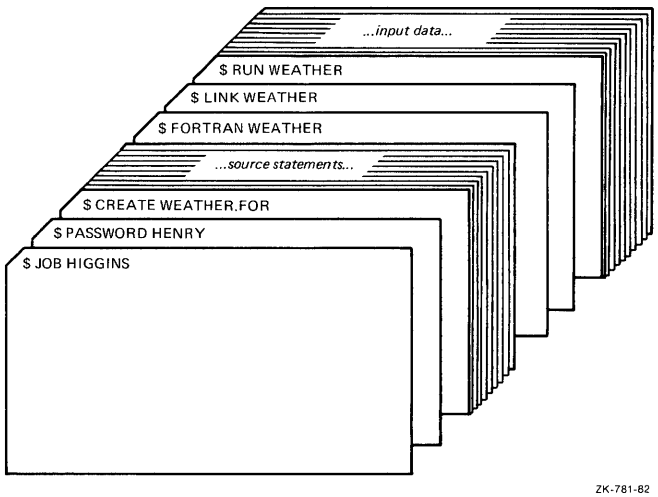
After you enter the CREATE command from the terminal, the system reads input lines into the sequential file A.DAT until CTRL/Z terminates the first input. The next set of input data is placed in the second file, B.DAT. Again, CTRL/Z terminates the input.

3

```
$ FILE = F$SEARCH("MEET.TXT")
$ IF FILE .EQS. ""
$ THEN CREATE MEET.TXT
  John, Residents in the apartment complex will hold their annual meeting
  this evening.  We hope to see you there, Regards, Elwood
$ ELSE TYPE MEET.TXT
$ ENDIF
$ EXIT
```

In this example, the command procedure searches the default disk and directory for the file MEET.TXT. If the command procedure determines that the file does not exist it creates a file named MEET.TXT using the CREATE command.

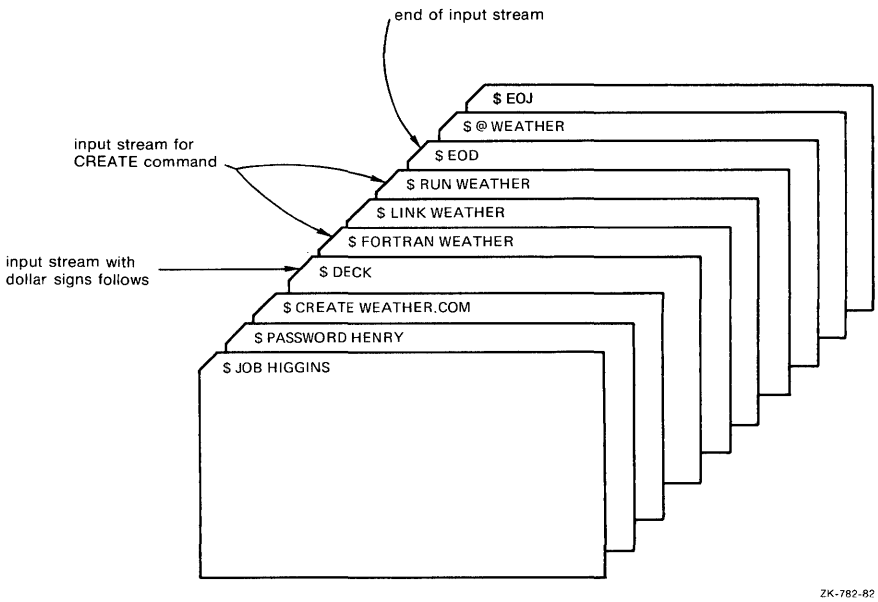
4



In this batch job example, the CREATE command creates a FORTRAN source file WEATHER.FOR. Records are read into that file until the system

encounters a dollar sign in the first position of the record $ FORTRAN WEATHER. The next commands compile, link, and run the file just created. Input data follows the RUN command.

5



This batch job example uses the CREATE command to create a command procedure from data in the input stream. The DECK command is required so that subsequent lines that begin with a dollar sign are not executed as commands, but are accepted as input records. The EOD command signals the end-of-file for the data records. Then the WEATHER procedure is executed with the @ (Execute Procedure) command.

CREATE/DIRECTORY

Creates one or more new directories or subdirectories. The /DIRECTORY qualifier is required.

Requires WRITE (W) access to the master file directory (MFD) to create a first-level directory. On a system volume, generally only users with a system UIC or the SYSPRV or BYPASS user privileges have WRITE access to the MFD to create a first-level directory.

Requires WRITE access to the lowest level directory that currently exists to create a subdirectory.

FORMAT

CREATE/DIRECTORY *directory-spec[,...]*

PARAMETER

directory-spec[,...]

Specifies the name of one or more directories or subdirectories to be created. The directory specification optionally can be preceded by a device name (and colon). The default is the current default directory. Wildcard characters are not allowed. When creating a subdirectory, separate the names of the directory levels with periods.

Note that it is possible to create a series of nested subdirectories with a single CREATE/DIRECTORY command. For example, [a.b.c] can be created, even though neither [a.b] nor [a] exists at the time the command is entered. Each subdirectory will be created, starting with the highest level and proceeding downwards.

DESCRIPTION

The CREATE/DIRECTORY command creates new directories as well as subdirectories. Special privileges are needed to create new first-level directories. (See the restrictions noted above.) Generally, users have sufficient privileges to create subdirectories in their own directories. Use the SET DEFAULT command to move from one directory to another.

QUALIFIERS

/LOG
/NOLOG (default)

Controls whether the CREATE/DIRECTORY command displays the directory specification of each directory after creating it.

/OWNER_UIC[=option]

Requires SYSPRV privilege for a UIC (user identification code) other than your own.

Specifies an owner UIC for the directory. The default is your UIC. You can specify the keyword PARENT in place of a UIC to mean the UIC of the parent (next-higher-level) directory. If a user with privileges creates a subdirectory, by default, the owner of the subdirectory will be the owner of the parent directory (or the owner of the Master File Directory, if creating a main level directory). If you do not specifiy the /OWNER_UIC qualifier when creating a directory, the command assigns ownership as follows: (1) if you specify the directory name in either alphanumeric or subdirectory format, the default is your UIC (unless you are privileged in which case the UIC defaults to the parent directory); (2) if you specify the directory in UIC format, the default is the specified UIC. Specify the UIC using standard UIC format as described in Section 8.1 of the *VMS DCL Concepts Manual.*

/PROTECTION=(code)

Specifies protection for the directory. Specify ownership as SYSTEM, OWNER, GROUP, or WORLD and protection as R (read), W (write), E (execute), or D (delete). The default protection is the protection of the parent directory (the next-higher level directory, or the master directory for top-level directories) minus any delete access.

If you are creating a first-level directory, then the next-higher-level directory is the MFD. (The protection of the MFD is established by the INITIALIZE command.)

See Section 8.1 of the *VMS DCL Concepts Manual* for more information on specifying protection code.

/VERSION_LIMIT=n

Specifies the number of versions of any one file that can exist in the directory. If you exceed the limit, the system deletes the lowest numbered version. A specification of 0 means no limit. The maximum number of versions allowed is 32,767. The default is the limit for the parent (next-higher-level) directory.

When you change the version limit setting, the new limit applies only to files created after the setting was changed. New versions of files created before the change are subject to the previous version limit.

/VOLUME=n

Requests that the directory file be placed on the specified relative volume of a multivolume set. By default, the file is placed arbitrarily within the multivolume set.

EXAMPLES

1 `$ CREATE/DIRECTORY/VERSION_LIMIT=2 $DISK1:[ACCOUNTS.MEMOS]`

In this example, the CREATE/DIRECTORY command creates a subdirectory named MEMOS in the ACCOUNTS directory on $DISK1. No more than two versions of each file can exist in the directory.

2

```
$ CREATE/DIRECTORY/PROTECTION=(SYSTEM:RWED,OWNER:RWED,GROUP,WORLD) -
_$[MALCOLM.SUB.HLP]
```

In this example, the CREATE/DIRECTORY command creates a subdirectory named [MALCOLM.SUB.HLP]. The protection on the subdirectory allows read, write, execute, and delete access for the system and owner categories, but prohibits all access for the group or world categories.

3

```
$ CREATE/DIRECTORY DISK2:[MALCOLM]
```

In this example, the CREATE/DIRECTORY command creates a directory named [MALCOLM] on the device DISK2. Special privileges are required to create a first-level directory.

4

```
$ CREATE/DIRECTORY [MALCOLM.SUB]
$ SET DEFAULT [MALCOLM.SUB]
```

In this example, the CREATE/DIRECTORY command creates a subdirectory named [MALCOLM.SUB]. This directory file is placed in the directory named [MALCOLM]. The command SET DEFAULT [MALCOLM.SUB] changes the current default directory to this subdirectory. All files subsequently created are cataloged in [MALCOLM.SUB].

5

```
$ CREATE/DIRECTORY [FRED.SUB1.SUB2.SUB3]
```

In this example, the CREATE/DIRECTORY command creates a top-level directory ([FRED]) and three subdirectories ([FRED.SUB1], [FRED.SUB1.SUB2], and [FRED.SUB1.SUB2.SUB3]).

CREATE/FDL

Invokes the Create/FDL Utility (CREATE/FDL) to use the specifications in an FDL file to create a new, empty data file. Use this utility to create a data file from a particular FDL specification. The /FDL qualifier is required. For a complete description of the Create/FDL Utility, including more information about the CREATE/FDL command and its qualifier, see the FDL Utility document in the *VMS File Definition Language Facility Manual*.

FORMAT **CREATE/FDL** *=fdl-file-spec [file-spec]*

CREATE/NAME_TABLE

Creates a new logical name table. The full command, CREATE/NAME_TABLE, is required.

FORMAT

CREATE/NAME_TABLE *table-name*

PARAMETER

table-name

Specifies a string of 1 to 31 characters that identifies the logical name table you are creating. The string can include alphanumeric characters, the dollar sign, and the underscore. This name is entered as a logical name in either the process directory logical name table (LNM$PROCESS_DIRECTORY) or the system directory logical name table (LNM$SYSTEM_DIRECTORY).

DESCRIPTION

The CREATE/NAME_TABLE command creates a new logical name table. The name of the table is contained within the LNM$PROCESS_DIRECTORY directory table if the table is process-private, and within the LNM$SYSTEM_DIRECTORY directory table if the table is shareable.

Every new table has a parent table, which determines whether the new table is process-private or shareable. To create a process-private table, use the /PARENT_TABLE qualifier to specify the name of a process-private table (the process directory table). To create a shareable table, specify the parent as a shareable table.

If you do not explicitly provide a parent table, the CREATE/NAME_TABLE command creates a process-private table whose parent is LNM$PROCESS_DIRECTORY; that is, the name of the table is entered in the process directory.

Every table has a size quota. The quota may either constrain the potential growth of the table or indicate that the table's size can be virtually unlimited. The description of the /QUOTA qualifier explains how to specify a quota.

To specify an access mode for the table you are creating, use the /USER_MODE, /SUPERVISOR_MODE, or /EXECUTIVE_MODE qualifiers. If you specify more than one of these qualifiers, only the last one entered is accepted. If you do not specify an access mode, then a supervisor mode table is created.

To delete a logical name table, use the DEASSIGN command, specify the name of the table you want to delete, and use the /TABLE qualifier to specify the directory table where the name of the table was entered.

QUALIFIERS

/ATTRIBUTES[=(keyword[,...])]

Specifies attributes for the logical name table. If you specify only one keyword, you can omit the parentheses. If you do not specify the /ATTRIBUTES qualifier, no attributes are set.

You can specify the following keywords for attributes:

CONFINE	Does not copy the table name or the logical names contained in the table into a spawned subprocess; used only when creating a private logical name table. If a table is created with the CONFINE attribute, all names subsequently entered into the table are also confined.
NO_ALIAS	No identical names (either logical names or names of logical name tables) may be created in an outer (less privileged) mode in the current directory. If you do not specify NO_ALIAS, then the table may be "aliased" by an identical name created in an outer access mode. Deletes any previously created identical table names in an outer access mode in the same logical name table directory.
SUPERSEDE	Creates a new table that supersedes any previous (existing) table that contains the name, access mode, and directory table that you specify. The new table is created regardless of whether the previous table exists. (If you do not specify the SUPERSEDE attribute, the new table is not created if the previous table exists.) If you specify or accept the default for the qualifier /LOG, you receive a message indicating the result.

/EXECUTIVE_MODE

Requires SYSNAM privilege.

Creates an executive mode logical name table. If you specify executive mode without having SYSNAM privilege, a supervisor mode logical name table is created.

/LOG (default)
/NOLOG

Controls whether or not an informational message is generated when the SUPERSEDE attribute is specified, or when the table already exists but the SUPERSEDE attribute is not specified. The default is /LOG; that is, the informational message is displayed.

/PARENT_TABLE=table

Requires EXECUTE (E) access to the parent table and SYSPRV privilege to create a shareable logical name table.

Specifies the name of the parent table. The parent table determines whether a table is private or shareable; it also determines the size quota of the table. If you do not specify a parent table, the default table is LNM$PROCESS_DIRECTORY. A shareable table has LNM$SYSTEM_DIRECTORY as its parent table. The parent table must have the same access mode or a higher-level access mode than the one you are creating.

/PROTECTION

Applies the specified protection to shareable name tables. The ownership categories are SYSTEM, OWNER, GROUP, WORLD; the access categories are R (READ), W (WRITE), E (EXECUTE) and D (DELETE). The default protection is (SYSTEM:RWED,OWNER:RWED,GROUP:,WORLD:) See Section 8.1 of the *VMS DCL Concepts Manual* for more information on specifying protection code.

Protection applies only to shareable logical name tables; it does not apply to process-private logical name tables.

/QUOTA=number-of-bytes

Specifies the size limit of the logical name table. The size of each logical name entered in the new table is deducted from this size limit. The new table's quota is statically subtracted from the parent table's quota holder. The parent table's quota holder is the first logical name table encountered when working upward in the table hierarchy that has an explicit quota and is therefore its own quota holder. If /QUOTA is not specified or the size limit is 0, the parent table's quota holder becomes the new table's quota holder and space is dynamically withdrawn from it whenever a logical name is entered in this new table. If you do not specify the /QUOTA qualifier, or if you specify /QUOTA=0, the table has unlimited quota.

/SUPERVISOR_MODE (default)

Creates a supervisor mode logical name table. If you do not specify a mode, a supervisor mode logical name table is created.

/USER_MODE

Creates a user mode logical name table. If you do not explicitly specify a mode, a supervisor mode logical name table is created.

EXAMPLES

1

```
$ CREATE/NAME_TABLE TEST_TAB
$ SHOW LOGICAL TEST_TAB
%SHOW-S-NOTRAN, no translation for logical name TEST_TAB
$ SHOW LOGICAL/TABLE=LNM$PROCESS_DIRECTORY TEST_TAB
```

In this example, the CREATE/NAME_TABLE command creates a new table called TEST_TAB. By default, the name of the table is entered in the process directory. The first SHOW LOGICAL command does not find the name TEST_TAB because it does not, by default, search the process directory table. You must use the /TABLE qualifier to request that the process directory be searched.

2

```
$ CREATE/NAME_TABLE/ATTRIBUTES=CONFINE EXTRA
$ DEFINE/TABLE=EXTRA MYDISK DISK4:
$ DEFINE/TABLE=LNM$PROCESS_DIRECTORY LNM$FILE_DEV -
_$ EXTRA, LNM$PROCESS, LNM$JOB, LNM$GROUP, LNM$SYSTEM
$ TYPE MYDISK:[COHEN]EXAMPLE1.LIS
```

This example creates a new logical name table called EXTRA that is created with the CONFINE attribute. Therefore, the EXTRA table and the names it contains will not be copied to subprocesses.

Next, the logical name MYDISK is placed into the table EXTRA. To use the name MYDISK in file specifications, you must make sure that the table EXTRA is searched when RMS parses file specifications. To do this, you can define a process-private version of the logical name LNM$FILE_DEV to include the name EXTRA as one of its equivalence strings. (The system uses LNM$FILE_DEV to determine the tables to search during logical name translation for device or file specifications, and will use the process-private version of the logical name before using the default system version.) After you define LNM$FILE_DEV, the system searches the following tables during logical name translation: EXTRA, your process table, your job table, your group table, and the system table. Now, you can use the name MYDISK in a file specification and the equivalence string DISK4 will be substituted.

DEALLOCATE

Makes an allocated device available to other processes (but does not deassign any logical name associated with the device).

FORMAT

DEALLOCATE *device-name[:]*

PARAMETER

device-name[:]

Name of the device to be deallocated. The device name can be a physical device name or a logical name. On a physical device name, the controller defaults to A and the unit to 0. Incompatible with the /ALL qualifier.

QUALIFIER

/ALL

Deallocates all devices currently allocated by your process. Incompatible with the device-name parameter.

EXAMPLES

1

```
$ DEALLOCATE  DMB1:
```

In this example, the DEALLOCATE command deallocates unit 1 of the RK06/RK07 devices on controller B.

2

```
$ ALLOCATE   MT:   TAPE
%DCL-I-ALLOC, _MTB1:  allocated
    .
    .
    .
$ DEALLOCATE  TAPE:
```

In this example, the ALLOCATE command requests that any magnetic tape drive be allocated and assigns the logical name TAPE to the device. The response to the ALLOCATE command indicates the successful allocation of the device MTB1. The DEALLOCATE command specifies the logical name TAPE to release the tape drive.

3

```
$ DEALLOCATE/ALL
```

In this example, the DEALLOCATE command deallocates all devices that are currently allocated.

DEASSIGN

Cancels logical name assignments made with the ALLOCATE, ASSIGN, DEFINE, or MOUNT command. The DEASSIGN command also deletes logical name tables created with the CREATE/NAME_TABLE command. Logical names in private tables are deleted automatically when your process terminates. All logical names in the job table and the job table itself are deleted when your process terminates. User mode logical names in the process table are deleted automatically when the next image exits. All other logical names in shareable tables remain unless explicitly deassigned. All names in descendant tables are deleted when the parent table logical name is deassigned.

FORMAT

DEASSIGN *[logical-name[:]]*

PARAMETER

logical-name[:]

Specifies the logical name to be deassigned. Logical names can have from 1 to 255 characters. If the logical name contains any characters other than alphanumerics, dollar signs, or underscores, enclose it in quotation marks. The logical-name parameter is required unless you use the /ALL qualifier.

If the logical-name parameter ends with a colon, the command interpreter ignores the colon. (Note that the ASSIGN and ALLOCATE commands remove a trailing colon, if present, from a logical name before placing the name in a logical name table.) If a colon is present in the logical name, you must type two colons in the logical-name parameter of the DEASSIGN command (for example, DEASSIGN FILE::).

To delete a logical name table, specify the table name as the logical name parameter. You must also use the /TABLE qualifier to indicate the logical name directory table where the table name is entered.

DESCRIPTION

The DEASSIGN command cancels a logical name assignment that was made with one of the following commands: ALLOCATE, ASSIGN, CREATE /NAME_TABLE, DEFINE, or MOUNT. You can use the /ALL qualifier with DEASSIGN to cancel all logical names in a specified table. If you use the /ALL qualifier and do not specify a table, then all names in the process table (except names created by the command interpreter) are deassigned; that is, all names entered at the indicated access mode or an outer access mode are deassigned.

To specify the logical name table from which you want to deassign a logical name, use the /PROCESS, /JOB, /GROUP, /SYSTEM, or /TABLE qualifiers. If you enter more than one of these qualifiers, only the last one entered is accepted. If entries exist for the specified logical name in more than one logical name table, the name is deleted from only the last logical name table specified on the command line. If you do not specify a logical name table, the default is /TABLE=LNM$PROCESS (or /PROCESS).

DEASSIGN

To specify the access mode of the logical name you want to deassign, use the /USER_MODE, /SUPERVISOR_MODE, or /EXECUTIVE_MODE qualifiers. If you enter more than one of these qualifiers, only the last one is accepted. If you do not specify a mode, the DEASSIGN command deletes a supervisor mode name. When you deassign a logical name, any identical names created with outer access modes in the same logical name table are also deleted.

You must have SYSNAM privilege to deassign an executive mode logical name. If you specify /EXECUTIVE_MODE and you do not have SYSNAM privilege, then the DEASSIGN command ignores the qualifier and attempts to deassign a supervisor mode logical name.

All process-private logical names and logical name tables are deleted when you log off the system. User mode entries within the process logical name table are deassigned when any image exits. The logical names in the job table, and the job table itself, are deleted when you log off the system.

Names in all other shareable logical name tables remain there until they are explicitly deassigned, regardless of whether they are user, supervisor, or executive mode names. You must have WRITE (W) access to a shareable logical name table to delete any name in that table.

If you delete a logical name table, all the logical names in the table are also deleted. Also, any descendant tables are deleted. To delete a shareable logical name table, you must have the user privilege SYSPRV or you must have DELETE (D) access to the table.

QUALIFIERS

/ALL

Deletes all logical names in the same or an outer (less privileged) access mode. If no logical name table is specified, the default is the process table, LNM$PROCESS. If you specify /ALL, you cannot enter a logical-name parameter.

/EXECUTIVE_MODE

Requires SYSNAM privilege to deassign executive mode logical names.

Deletes only entries that were created in the specified mode or an outer (less privileged) mode. If you do not have SYSPRV privilege for executive mode, a supervisor mode operation is assumed.

/GROUP

Requires GRPNAM or SYSPRV privilege to delete entries from the group logical name table.

Indicates that the specified logical name is in the group logical name table. The /GROUP qualifier is synonymous with /TABLE=LNM$GROUP.

/JOB

Indicates that the specified logical name is in the jobwide logical name table. The /JOB qualifier is synonymous with /TABLE=LNM$JOB. If you do not explicitly specify a logical name table, the default is /PROCESS.

You should not deassign jobwide logical name entries that were made by the system at login time, for example, SYS$LOGIN, SYS$LOGIN_DEVICE, and SYS$SCRATCH. However, if you assign new equivalence names for these logical names (that is, create new logical names in outer access modes), you can deassign the names you explicitly created.

/PROCESS (default)

Indicates that the specified logical name is in the process logical name table. The /PROCESS qualifier is synonymous with /TABLE=LNM$PROCESS.

You cannot deassign logical name table entries that were made by the command interpreter, for example, SYS$INPUT, SYS$OUTPUT, and SYS$ERROR. However, if you assign new equivalence names for these logical names (that is, you create new logical names in outer access modes), you can deassign the names you explicitly created.

/SUPERVISOR_MODE (default)

Deletes entries in the specified logical name table that were created in supervisor mode. If you specify the /SUPERVISOR_MODE qualifier, the DEASSIGN command also deassigns user mode entries with the same name.

/SYSTEM

Requires SYSNAM or SYSPRV privilege to delete entries from the system logical name table.

Indicates that the specified logical name is in the system logical name table. The /SYSTEM qualifier is synonymous with /TABLE=LNM$SYSTEM.

/TABLE=name

Requires WRITE (W) access to the table to delete a shareable logical name. Requires SYSPRV or DELETE (D) access to delete a shareable logical name table.

Specifies the table from which the logical name is to be deleted. Defaults to LNM$PROCESS. The table can be the process, group, job, or system table, one of the directory tables, or the name of a user-created table. (The process, job, group, and system logical name tables should be referred to by the logical names LNM$PROCESS, LNM$JOB, LNM$GROUP, and LNM$SYSTEM, respectively.)

The /TABLE qualifier also can be used to delete a logical name table. To delete a process-private table, enter the following command:

```
$ DEASSIGN/TABLE=LNM$PROCESS_DIRECTORY table-name
```

To delete a shareable table, enter the following command:

```
$ DEASSIGN/TABLE=LNM$SYSTEM_DIRECTORY table-name
```

To delete a shareable logical name table, you must have DELETE (D) access to the table or WRITE (W) access to the directory table in which the name of the shareable table is cataloged.

If you do not explicitly specify the /TABLE qualifier, the default is /TABLE=LNM$PROCESS (or /PROCESS).

/USER_MODE

Deletes entries in the process logical name table that were created in user mode. If you specify the /USER_MODE qualifier, the DEASSIGN command can deassign only user mode entries.

DEASSIGN

EXAMPLES

1
```
$ DEASSIGN MEMO
```

The DEASSIGN command in this example deassigns the process logical name MEMO.

2
```
$ DEASSIGN/ALL
```

The DEASSIGN command in this example deassigns all process logical names that were created in user and supervisor mode. This command does not, however, delete the names that were placed in the process logical name table in executive mode by the command interpreter (for example, SYS$INPUT, SYS$OUTPUT, SYS$ERROR, SYS$DISK, and SYS$COMMAND).

3
```
$ DEASSIGN/TABLE=LNM$PROCESS_DIRECTORY TAX
```

The DEASSIGN command in this example deletes the logical name table TAX, and any descendant tables. When you delete a logical name table, you must specify either /TABLE=LNM$PROCESS_DIRECTORY or /TABLE=LNM$SYSTEM_DIRECTORY, because the names of all tables are contained in these directories.

4
```
$ ASSIGN USER_DISK:   COPY
$ DEASSIGN COPY
```

The ASSIGN command in this example equates the logical name COPY with the device USER_DISK and places the names in the process logical name table. The DEASSIGN command deletes the logical name.

5
```
$ DEFINE SWITCH:  TEMP
$ DEASSIGN SWITCH::
```

The DEFINE command in this example places the logical name SWITCH: in the process logical name table. The trailing colon is retained as part of the logical name. Two colons are required on the DEASSIGN command to delete this logical name because the DEASSIGN command removes one trailing colon, and the other colon is needed to match the characters in the logical name.

6
```
$ ASSIGN/TABLE=LNM$GROUP DBA1: GROUP_DISK
$ DEASSIGN/PROCESS/GROUP GROUP_DISK
```

The ASSIGN command in this example places the logical name GROUP_DISK in the group logical name table. The DEASSIGN command specifies conflicting qualifiers; because the /GROUP qualifier is last, the name is successfully deassigned.

7
```
$ ASSIGN DALLAS::USER_DISK:   DATA
    .
    .
    .
$ DEASSIGN DATA
```

The ASSIGN command in this example associates the logical name DATA with the device specification USER_DISK on remote node DALLAS. Subsequent references to the logical name DATA result in references to the disk on the remote node. The DEASSIGN command cancels the logical name assignment.

DEASSIGN/QUEUE

Deassigns a logical queue from a printer or terminal queue and stops the logical queue. The DEASSIGN/QUEUE command is the complement of the ASSIGN/QUEUE command.

Requires OPER privilege or EXECUTE access to the queue. Cannot be used with batch queues.

FORMAT

DEASSIGN/QUEUE *logical-queue-name[:]*

PARAMETER

logical-queue-name[:]
Specifies the name of the logical queue that you want to deassign from a specific printer or terminal queue.

DESCRIPTION

Once you enter the DEASSIGN/QUEUE command, the jobs in the logical queue remain pending until the queue is reassigned to another printer queue or device with the ASSIGN/QUEUE command.

EXAMPLE

```
$ ASSIGN/QUEUE LPAO  ASTER
   .
   .
   .
$ DEASSIGN/QUEUE  ASTER
$ ASSIGN/MERGE LPBO  ASTER
```

The ASSIGN/QUEUE command in this example associates the logical queue ASTER with the print queue LPA0. Later, you deassign the logical queue with the DEASSIGN/QUEUE command. The ASSIGN/MERGE command reassigns the jobs from ASTER to the print queue LPB0.

DEBUG

Invokes the VMS Debugger after program execution is interrupted by CTRL/Y, but only if the /NOTRACEBACK qualifier was not specified with the LINK command when the program was linked. For a complete description of the VMS Debugger, including more information about the DEBUG command, see the *VMS Debugger Manual*.

FORMAT DEBUG

DECK

Marks the beginning of an input stream for a command or program. The DECK command is required in command procedures when the first nonblank character in any data record in the stream is a dollar sign.

Can be used only after a request to execute a command or program that requires input data.

FORMAT

DECK

DESCRIPTION

The DECK command marks the data that follows it as input for a command or program. This command is required in command procedures when the first nonblank character in any data record in the input stream is a dollar sign.

The DECK command must be preceded by a dollar sign; the dollar sign must be in the first character position (column 1) of the input record.

The DECK command defines an end-of-file indicator only for a single data stream. Using the DECK command enables you to place data records beginning with dollar signs in the input stream. You can place one or more sets of data in the input stream following a DECK command, if each is terminated by an end-of-file indicator.

After an end-of-file indicator specified with the /DOLLARS qualifier is encountered, the end-of-file indicator is reset to the default, that is, to any record beginning with a dollar sign. The default is also reset if an actual end-of-file occurs for the current command level.

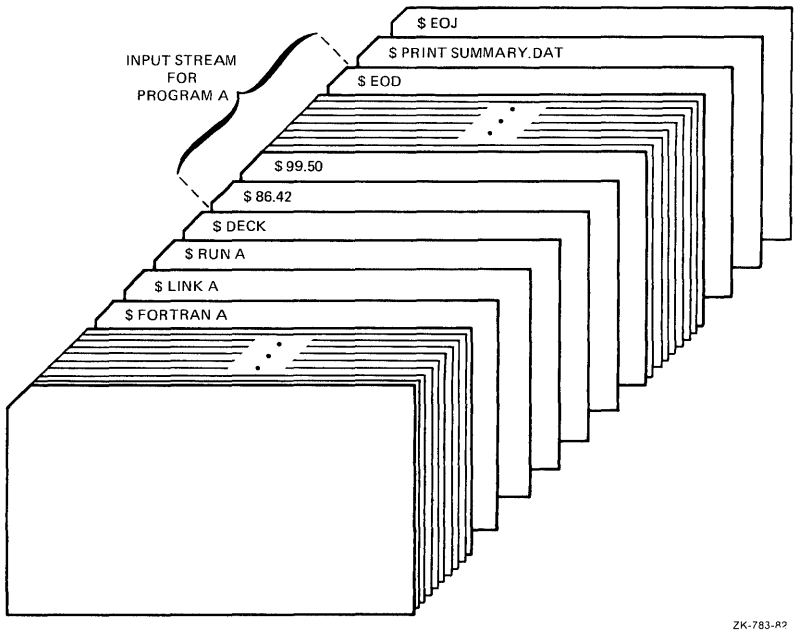
QUALIFIER

/DOLLARS[=string]

Sets the end-of-file indicator to the specified string of 1 through 15 characters. Specify a string if the input data contains one or more records beginning with the string $EOD. Enclose the string in quotation marks if it contains literal lowercase letters, multiple blanks, or tabs. If you do not specify /DOLLARS, or if you specify /DOLLARS without specifying a string, you must use the EOD command to signal the end-of-file.

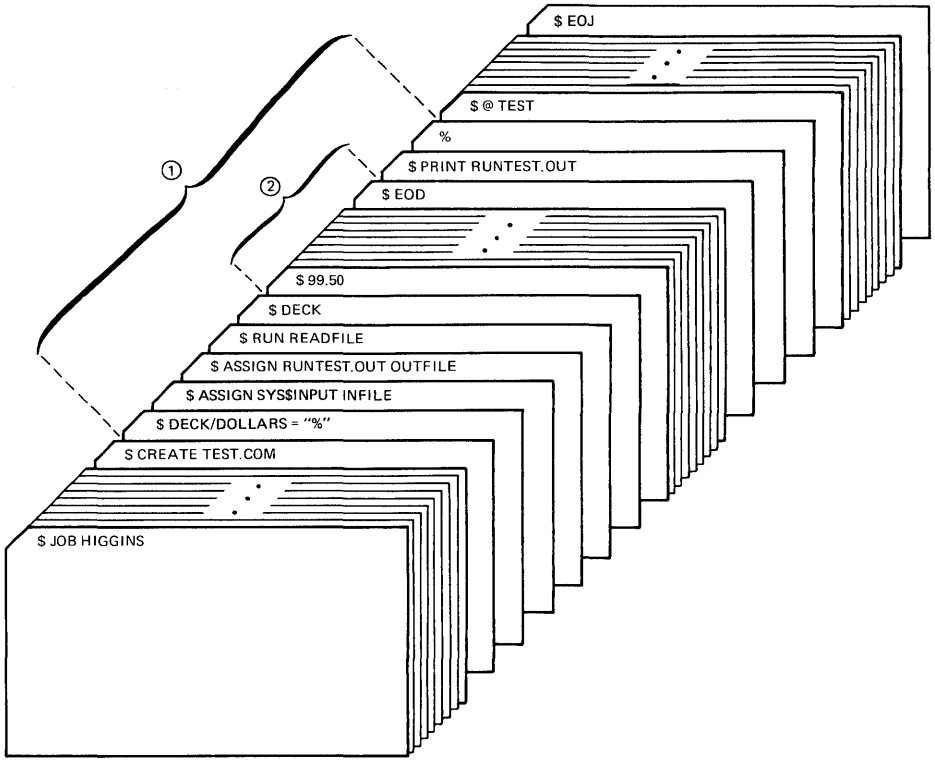
EXAMPLES

1



In this example, the FORTRAN and LINK commands compile and link program A. When the program is run, any data the program reads from the logical device SYS$INPUT is read from the command stream. The DECK command indicates that the input stream can contain dollar signs in column 1 of the record. The EOD command signals end-of-file for the data.

2



The CREATE command in this example creates the command procedure file TEST.COM from lines entered into the input stream. The DECK/DOLLARS command indicates that the percent sign character is the end-of-file indicator for the CREATE command. This allows the string $EOD to be read as an input record, signaling the end of the input for the RUN command.

DEFINE

Associates equivalence names with a logical name. If you specify an existing logical name, the new equivalence names replace the existing equivalence name.

FORMAT

DEFINE *logical-name equivalence-name[,...]*

PARAMETERS

logical-name

Specifies the logical name string, which is a character string containing from 1 to 255 characters. If the logical name is to be entered into the process or system directory logical name tables (LNM$PROCESS_DIRECTORY, LNM$SYSTEM_DIRECTORY), then the name may only have from 1 to 31 alphanumeric characters (including the dollar sign and underscore).

If you specify a colon at the end of a logical name, the DEFINE command saves the colon as part of the logical name. (This is in contrast to the ASSIGN command, which removes the colon before placing the name in a logical name table.) By default, the logical name is placed in the process logical name table.

If the string contains any characters other than uppercase alphanumerics, the dollar sign, or the underscore character, enclose the string in quotation marks (" "). Use two consecutive quotation marks ("") to denote an actual quotation mark. Note that if you enclose a name in quotation marks, the case of alphabetic characters is preserved.

equivalence-name[,...]

Specifies a character string containing from 1 to 255 characters. If the string contains any characters other than uppercase alphanumerics, the dollar sign, or the underscore character, enclose the string in quotation marks. Use two consecutive quotation marks ("") to denote an actual quotation mark. Specifying more than one equivalence name for a logical name creates a search list.

When you specify an equivalence name that will be used as a file specification, you must include the punctuation marks (colons, brackets, periods) that would be required if the equivalence name were used directly as a file specification. Therefore, if you specify a device name as an equivalence name, you must terminate the equivalence name with a colon.

The DEFINE command allows you to assign the same logical name to more than one equivalence name. For example, you can use the same logical name to access different directories on different disks, or to access different files in different directories. When you specify more than one equivalence name for a logical name, you create a search list. See Section 4.7 of the *VMS DCL Concepts Manual* for more information on search lists.

DESCRIPTION

The DEFINE command creates an entry in a logical name table by defining a logical name to stand for one or more equivalence names. An equivalence name can be a device name, another logical name, a file specification, or any other string.

To specify the logical name table where you want to enter a logical name, use the /PROCESS, /GROUP, /SYSTEM, /JOB, or /TABLE qualifier. If you enter more than one of these qualifiers, only the last one entered is accepted. If you do not specify a table, the default is /TABLE=LNM$PROCESS (or /PROCESS).

To specify the access mode of the logical name you are creating, use the /USER_MODE, /SUPERVISOR_MODE, or /EXECUTIVE_MODE qualifiers. If you enter more than one of these qualifiers, only the last one entered is accepted. If you do not specify an access mode, a supervisor mode name is created. You can create a logical name in the same mode as the table in which you are placing the name, or in an outer mode. (User mode is the outermost mode; executive mode is the innermost mode.)

You can enter more than one logical name with the same name in the same table, as long as each name has a different access mode. (However, if an existing logical name within a table has the NO_ALIAS attribute, you cannot use the same name to create a logical name in an outer mode in this table.)

If you create a logical name with the same name, in the same table, and in the same mode as an existing name, the new logical name assignment replaces the existing assignment.

You can also use the ASSIGN command to create logical names. To delete a logical name from a table, use the DEASSIGN command.

Note: Avoid assigning a logical name that matches the file name of an executable image in SYS$SYSTEM:. Such an assignment prohibits you from invoking that image.

For additional information on how to create and use logical names, see Chapter 4 of the *VMS DCL Concepts Manual*.

QUALIFIERS

/EXECUTIVE_MODE

Requires SYSNAM privilege to create an executive mode logical name.

Creates an executive mode logical name in the specified table.

If you specify the /EXECUTIVE_MODE qualifier and you do not have SYSNAM, the DEFINE command ignores the qualifier and creates a supervisor mode logical name. The mode of the logical name must be the same or less privileged than the mode of the table in which you are placing the name.

/GROUP

Requires GRPNAM or SYSPRV privilege to place a name in the group logical name table.

Places the logical name in the group logical name table. Other users who have the same group number in their UICs (user identification codes) can access the logical name. The /GROUP qualifier is synonymous with /TABLE=LNM$GROUP.

/JOB

Places the logical name in the jobwide logical name table. All processes in the same job tree as the process that created the logical name can access the logical name. The /JOB qualifier is synonymous with /TABLE=LNM$JOB.

/LOG (default)
/NOLOG

Displays a message when a new logical name supersedes an existing name.

/NAME_ATTRIBUTES[=(keyword[,...])]

Specifies attributes for a logical name. By default, no attributes are set. Possible keywords are as follows:

CONFINE	The logical name is not copied into a spawned subprocess. This qualifier is relevant only for logical names in a private table. The logical name inherits the CONFINE attribute from the logical name table where it is entered; if the logical name table is "confined", then all names in the table are "confined".
NO_ALIAS	A logical name cannot be duplicated in the specified table in a less privileged access mode; any previously created identical names in an outer (less privileged) access mode within the specified table are deleted.

If you specify only one keyword, you can omit the parentheses. Only the attributes you specify are set.

/PROCESS (default)

Places the logical name in the process logical name table. The /PROCESS qualifier is synonymous with /TABLE=LNM$PROCESS.

/SUPERVISOR_MODE (default)

Creates a supervisor mode logical name in the specified table. The mode of the logical name must be the same as or less privileged than the mode of the table in which you are placing the name.

/SYSTEM

Requires SYSNAM or SYSPRV privilege to place a name in the system logical name table.

Places the logical name in the system logical name table. All system users can access the logical name. The /SYSTEM qualifier is synonymous with /TABLE=LNM$SYSTEM.

/TABLE=name

Requires WRITE (W) access to the table to specify the name of a shareable logical name table.

Specifies the name of the logical name table in which the logical name is to be entered. You can use the /TABLE qualifier to specify a user-defined logical name table (created with the CREATE/NAME_TABLE command); to specify the process, job, group, or system logical name tables; or to specify the process or system logical name directory tables.

If you specify the table name using a logical name that has more than one translation, the logical name is placed in the first table found. For example, if you specify DEFINE/TABLE=LNM$FILE_DEV and LNM$FILE_DEV is equated to LNM$PROCESS, LNM$JOB, LNM$GROUP, and LNM$SYSTEM, then the logical name is placed in LNM$PROCESS.

The default is /TABLE=LNM$PROCESS (or /PROCESS).

/TRANSLATION_ATTRIBUTES[=(keyword[,...])]

Equivalence-name qualifier.

Specifies one or more attributes that modify an equivalence string of the logical name. Possible keywords are as follows:

CONCEALED	Indicates that the equivalence string is the name of a concealed device. When a concealed device name is defined, the system displays the logical name, rather than the equivalence string, in messages that refer to the device.
TERMINAL	Logical name translation should terminate with the current equivalence string; indicates that the equivalence string should not be translated iteratively.

If you specify only one keyword, you can omit the parentheses. Only the attributes you specify are set.

Note that different equivalence strings of a logical name can have different translation attributes.

/USER_MODE

Creates a user mode logical name in the specified table.

User mode logical names created within the process logical name tables are used for the execution of a single image; for example, you can create a user mode logical name to allow an image executing in a command procedure to redefine SYS$INPUT. User mode entries are deleted from the process logical name table when any image executing in the process exits (that is, after a DCL command or user program that executes an image completes execution).

EXAMPLES

1
```
$ DEFINE MEMO $DISK1:[ACCOUNTS.MEMO]
```

In this example, the DEFINE command defines the logical name MEMO as equivalent to the partial file specification $DISK1:[ACCOUNTS.MEMO].

2
```
$ DEFINE/USER_MODE TM1 $DISK1:[ACCOUNTS.MEMOS]WATER.TXT
```

In this example, the DEFINE command defines TM1 as equivalent to a file specification. After the next image runs, the logical name TM1 is automatically deassigned.

3
```
$ DEFINE PROCESS_NAME  LIBRA
$ RUN WAKE
```

In this example, the DEFINE command places the logical name PROCESS_NAME in the process logical name table with an equivalence name of LIBRA. The logical name is created in supervisor mode. The program WAKE translates the logical name PROCESS_NAME to perform some special action on the process named LIBRA.

4
```
$ DEFINE TEMP:  XXX1:
   .
   .
   .
$ DEASSIGN TEMP::
```

In this example, the DEFINE command creates an equivalence name for the logical name TEMP: and places the name in the process logical name table. The colon is retained as part of the logical name. The DEASSIGN command deletes the logical name. Note that two colons are required on the logical name in the DEASSIGN command. One colon is deleted by the DEASSIGN command. The other colon is kept as part of the logical name.

5
```
$ DEFINE PORTLAND PRTLND::YYY0:[DECNET.DEMO.COM]
```

In this example, the DEFINE command places the logical name PORTLAND in the process logical name table with an equivalence name of PRTLND::YYY0:[DECNET.DEMO.COM]. Subsequent references to the logical name PORTLAND result in the correspondence between the logical name PORTLAND and the node, disk, and subdirectory specified.

6
```
$ DEFINE LOCAL "BOSTON""JOHN_SMITH JKS""::"
```

In this example, the DEFINE command places the logical name LOCAL in the process logical name table with a remote node equivalence name of BOSTON"JOHN_SMITH JKS"::. To satisfy conventions for local DCL command string processing, you must use three sets of quotation marks. The quotation marks ensure that access control information is enclosed in one set of quotation marks in the equivalence name.

7
```
$ DEFINE MYDISK XXX0:[MYDIR], YYY0:[TESTDIR]
```

In this example, the DEFINE command places the logical name MYDISK in the process logical name table with two equivalence names: XXX0:[MYDIR] and YYY0:[TESTDIR].

8
```
$  CREATE/NAME_TABLE TABLE1
$  DEFINE/TABLE=LNM$PROCESS_DIRECTORY LNM$FILE_DEV -
_$ TABLE1,LNM$PROCESS,LNM$JOB,LNM$GROUP,LNM$SYSTEM
$  DEFINE/TABLE=TABLE1 -
_$ /TRANSLATION_ATTRIBUTES=CONCEALED   WORK_DISK DBA1:
```

In this example, the CREATE/NAME_TABLE command creates the process private logical name table TABLE1.

The first DEFINE command ensures that TABLE1 is searched first in any logical name translation of a device or file specification (because TABLE1 is the first item in the equivalence string for the logical name LNM$FILE_DEV, which determines the default search sequence of logical name tables whenever a device or file specification is translated).

The second DEFINE command assigns the logical name WORK_DISK to the physical device DBA1 and places the name in TABLE1. The logical name has the concealed attribute. Therefore, the logical name WORK_DISK is displayed in system messages.

9

```
$ CREATE/NAME_TABLE SPECIAL
$ DEFINE/TABLE=LNM$PROCESS_DIRECTORY LNM$FILE_DEV -
_$ SPECIAL,LNM$PROCESS,LNM$JOB,LNM$GROUP,LNM$SYSTEM
$ DEFINE/TABLE=LNM$PROCESS_DIRECTORY TAB SPECIAL
$ DEFINE/TABLE=TAB REPORT [CHELSEA]STORES
$ SHOW LOGICAL/TABLE=SPECIAL REPORT
  "REPORT" = "[CHELSEA]STORES" (SPECIAL)
```

In this example, the CREATE/NAME_TABLE command is used to create a new logical name table called SPECIAL. This table is defined in the process directory, LNM$PROCESS_DIRECTORY.

The first DEFINE command ensures that SPECIAL is searched first in any logical name translation of a device or file specification (because SPECIAL is the first item in the equivalence string for the logical name LNM$FILE_DEV, which determines the default search sequence of logical name tables whenever a device or file specification is translated). The logical name LNM$FILE_DEV is placed in the process directory, LNM$PROCESS_DIRECTORY.

With the next DEFINE command, a new logical name, TAB, is defined. TAB translates to the string SPECIAL, which identifies a logical name table. You must define TAB in the process directory because it translates iteratively to a logical name table.

Next, the logical name REPORT is placed into the logical name table TAB. Because TAB translates to the table SPECIAL, the name REPORT is entered into SPECIAL table. The SHOW LOGICAL command verifies that the name REPORT has been entered into the table SPECIAL.

Note that you can redefine TAB so it translates to a different table. Therefore, if you run different programs that use the name TAB as a table name, you can change the actual tables where the names are entered or referenced.

DEFINE/CHARACTERISTIC

Assigns a numeric value to a queue characteristic. The characteristic is created if it does not exist. If a value is already assigned to the characteristic, DEFINE/CHARACTERISTIC alters the assignment of that existing characteristic. The /CHARACTERISTIC qualifier is required. Used in conjunction with the /CHARACTERISTIC qualifier of the PRINT command.

Requires OPER privilege.

FORMAT

DEFINE/CHARACTERISTIC *characteristic-name*
characteristic-number

PARAMETERS

characteristic-name

Assigns a name to the characteristic being defined, which can be the name of an existing characteristic or a string of 1 to 31 characters that defines a new characteristic. The character string can include any uppercase and lowercase letters, digits, the dollar sign ($), and the underscore (_), and must include at least one alphabetic character.

characteristic-number

Assigns a number in the range 0 through 127 to the characteristic being defined.

DESCRIPTION

The system manager or operator uses the DEFINE/CHARACTERISTIC command to assign a name and number to a particular characteristic for queues in the system. Characteristics can refer to any attribute of a print or batch job that is meaningful for your environment. The name and number of a characteristic are arbitrary, but they must be unique for that characteristic. You can use the DEFINE/CHARACTERISTIC command to add a new characteristic or change the number of a previously defined characteristic. Use the SHOW QUEUE/CHARACTERISTICS command to find out what characteristics are currently defined for the system. The DELETE/CHARACTERISTIC command deletes a previously defined characteristic.

When queues are initialized or started, you can use either characteristic names or numbers with the /CHARACTERISTICS qualifier to specify characteristics to be associated with the queue. Similarly, when users enter the PRINT or SUBMIT command with the /CHARACTERISTICS qualifier, they can use either the characteristic name or number to specify which queue characteristics must match before the job is executed.

The SET QUEUE command changes the characteristics of a queue. To change the physical setup of the queue, use the STOP/QUEUE/NEXT command to stop the queue. Then change the setup and enter the START/QUEUE command with the new characteristics specified.

When users include the /CHARACTERISTICS qualifier with a PRINT or SUBMIT command, all the characteristics they specify must also be specified for the queue executing the job. If not, the job remains pending in the queue until the queue characteristics are changed or the users delete the entry with the DELETE/ENTRY command. Users need not specify every characteristic of a queue with a PRINT or SUBMIT command as long as the ones they specify are a subset of the characteristics set for that queue. The job will also run if no characteristics are specified.

The SHOW QUEUE/CHARACTERISTICS command displays the characteristics available on the system. Use the SHOW QUEUE/FULL command to find out which characteristics have been specified for a particular queue.

EXAMPLE

```
$ DEFINE/CHARACTERISTIC REDINK 3
```

The DEFINE command in this example defines the characteristic REDINK with the number 3. When a user enters the command PRINT /CHARACTERISTICS=REDINK (or PRINT /CHARACTERISTICS=3), the job is printed only if the printer queue has been established with the REDINK or 3 characteristic.

DEFINE/FORM

Assigns a numeric value to a print form name and defines the type of physical paper stock. If a value is already assigned to the form name, DEFINE/FORM alters the definition of the existing form. The /FORM qualifier is required. Used in conjunction with the /FORM qualifier of the PRINT command.

Requires OPER privilege.

FORMAT

DEFINE/FORM *form-name form-number*

PARAMETERS

form-name

Assigns a name to the form being defined. The form name can be the name of an existing form type or a string of 1 to 31 characters that defines a new form type. The character string can include any uppercase and lowercase letters, digits, the dollar sign ($), and the underscore (_), and must include at least one alphabetic character.

form-number

Assigns a number in the range 0 through 999 to the form being defined. The DEFAULT form, which is automatically defined when the system is bootstrapped, is assigned number 0.

DESCRIPTION

The system manager or operator uses the DEFINE/FORM command to assign a name and number to a type of paper stock or printing area for use with printer or terminal queues. When a new queue file is created, the system defines a form named DEFAULT with a form number of zero and all the default attributes.

The DEFINE/FORM qualifiers specify the area for printing. The /LEFT, /RIGHT, and /WIDTH qualifiers determine the number of characters per line. Using the /RIGHT and /WIDTH qualifiers, you can affect the point at which lines of text in the file will wrap. (These qualifiers cannot be used for filling or formatting the text, however.)

You can also use the DEFINE/FORM command to specify different types of paper stock. The /DESCRIPTION qualifier enables you to describe more fully the form name.

When a printer or terminal queue is initialized, you can use either the form name or number to specify the form for the queue. Similarly, when users enter a PRINT command with the /FORM qualifier, they can use either the form name or number to specify which form they want. The default form is number 0.

To change the form type of a queue, stop the queue using the STOP/QUEUE /NEXT command, change the physical form, and then restart the queue specifying the appropriate form with the START/QUEUE command.

When you include the /FORM qualifier with a PRINT command, the form you specify must match the one specified for the queue executing the job. If not, the job remains pending in the queue until the queue characteristics are changed or you delete the entry with the DELETE/ENTRY command. If you omit the /FORM qualifier from your PRINT command, your job is printed using the default form definition.

The SHOW QUEUE/FORM command displays the forms available on the system. Use the SHOW QUEUE/FULL command to find out what form has been specified for a particular queue.

QUALIFIERS

/DESCRIPTION=string

A string of up to 255 characters used to describe the form more specifically. The default string is the specified form name.

The string can be used to define the form type more specifically. For example, if you have form names such as LETTER1, LETTER2, and LETTER3, the /DESCRIPTION qualifier could be used to let the users and operators know that LETTER1 refers to the standard corporate letterhead paper (8.5 x 11), LETTER2 refers to the smaller corporate letterhead paper (6 x 9), and LETTER3 refers to the president's personalized letterhead paper.

If the string contains alphanumeric, underscore, or dollar sign characters, it must be enclosed in quotation marks (").

/LENGTH=n

Specifies the physical length of a form page in lines. The default page length is 66 lines, which assumes a standard page length of 11 inches with 6 lines of print per inch. The n parameter must be a positive integer greater than 0 and not more than 255.

The print symbiont sets the page length of the device equal to the form length. This enables the driver to compute the number of line feeds for devices lacking mechanical form feed.

/MARGIN=(option[,...])

Specifies one or more of the four margin options: BOTTOM, LEFT, RIGHT, and TOP.

BOTTOM=n — Specifies the number of blank lines between the end of the print image area and the end of the physical page; the value of n must be between 0 and the value of the /LENGTH parameter. The default value is 6, which generally means a one-inch bottom margin.

LEFT=n — Specifies the number of blank columns between the leftmost printing position and the print image area; the value of n must be between 0 and the value of the /WIDTH parameter. The default is 0, which means that the print image area starts as far to the left of the paper as the printer can go.

RIGHT=n — Specifies the number of blank columns between the /WIDTH parameter and the image area; the value of n must be between 0 and the value of the /WIDTH parameter. When determining the /RIGHT parameter, start at the /WIDTH value and count to the left. The default value is 0, which means that the print image extends as far to the right as the /WIDTH value.

TOP=n — Specifies the number of blank lines between the top of the physical page and the top of the print image; the value of n must be between 0 and the value of the /LENGTH parameter. The default value is 0, which generally means that there is no top margin.

/PAGE_SETUP=(module[,...])
/NOPAGE_SETUP (default)

Specifies one or more modules that set up the device before every page. The modules are located in the device control library. When a new page is detected, the system extracts the appropriate modules from the device control library and copies them to the printer before the page is printed.

/SETUP=(module[,...])

Specifies one or more modules in the device control library that set up the device appropriately for the specified form. When the form is mounted, the system extracts the specified module from the device control library and copies it to the printer before the file is printed.

/SHEET_FEED
/NOSHEET_FEED (default)

Specifies that print jobs pause at the end of every physical page so that a new sheet of paper can be inserted.

/STOCK=string

Specifies the type of paper stock to be associated with the form. The string parameter can be a string of 1 to 31 characters, including the dollar sign, underscore, and all alphanumeric characters. The default is the form name. If you specify the /STOCK qualifier you must specify the name of the stock to be associated with the form. If you do not specify the /STOCK qualifier, the name of the stock will be the same as the name of the form.

You can create any string that you want. However, when you are creating forms with the same stock, be sure that the /STOCK string is identical in all the DEFINE/FORM commands that refer to the same type of paper.

This qualifier is useful when you have several forms that use the same paper stock, but differ in other ways, such as margin specifications, wrapping, or page dimension. The system changes from one form to another automatically if those forms have an identical /STOCK qualifier. If the /STOCK qualifiers are different, stop the queue, change the form, and restart the queue to print on another stock.

/TRUNCATE (default)
/NOTRUNCATE

Discards any characters that exceed the current line length (specified by /WIDTH and /MARGIN=RIGHT). /TRUNCATE is incompatible with the /WRAP qualifier; the /TRUNCATE qualifier forces /NOWRAP. If you specify both /NOTRUNCATE and /NOWRAP, the printer prints as many characters on a line as possible. This combination of qualifiers is useful for some types of graphics output.

/WIDTH=n

Specifies the physical width of the paper in terms of columns or character positions. The n parameter must be an integer from 0 through 65,535; the default value is 132.

Any lines exceeding this value wrap if /WRAP is in effect or truncated if /TRUNCATE is in effect. (If both /NOTRUNCATE and /NOWRAP are in effect, lines print as far as possible.)

The /MARGIN=RIGHT qualifier overrides the /WIDTH qualifier when determining when to wrap lines of text.

/WRAP
/NOWRAP (default)

Causes lines that exceed the current line length (specified by /WIDTH and /MARGIN=RIGHT) to wrap onto the next line. /WRAP is incompatible with the /TRUNCATE qualifier; the /WRAP qualifier forces /NOTRUNCATE . If you specify both /NOWRAP and /NOTRUNCATE, the printer prints as many characters on a line as possible. This combination of qualifiers is useful for some types of graphics output.

EXAMPLE

```
$ DEFINE/FORM /MARGIN=(TOP=6,LEFT=10) CENTER 3
```

The DEFINE/FORM command in this example defines the form CENTER to have a top margin of 6 and a left margin of 10. The defaults remain in effect for both bottom margin (6) and right margin (0). The form is assigned the number 3.

DEFINE/KEY

Associates an equivalence string and a set of attributes with a key on the terminal keyboard. The /KEY qualifier is required.

FORMAT

DEFINE/KEY *key-name equivalence-string*

PARAMETERS

key-name

Specifies the name of the key that you are defining. The following table lists the key names in column one. The remaining three columns indicate the key designations on the keyboards of the three different types of terminals that allow key definitions. All definable keys on VT52 terminals are located on the numeric keypad. On VT100-series terminals, you can define the LEFT and RIGHT arrow keys as well as all the keys on the numeric keypad. On terminals with LK201 keyboards, three types of keys can be defined : (1) keys on the numeric keypad, (2) keys on the editing keypad (except the UP and DOWN arrow keys), and (3) keys on the function key row across the top of the keyboard. (Note that you cannot define function keys F1 through F5.)

Key-name	LK201	VT100-series	VT52
PF1	PF1	PF1	[blue]
PF2	PF2	PF2	[red]
PF3	PF3	PF3	[gray]
PF4	PF4	PF4	- -
KP0, KP1, ..., KP9	0, 1, ..., 9	0, 1, ..., 9	0, 1, ..., 9
PERIOD	.	.	.
COMMA	,	,	n/a
MINUS	-	-	n/a
ENTER	Enter	ENTER	ENTER
LEFT	←	←	←
RIGHT	→	→	→
Find (E1)	Find	- -	- -
Insert Here (E2)	Insert Here	- -	- -
Remove (E3)	Remove	- -	- -
Select (E4)	Select	- -	- -
Prev Screen (E5)	Prev Screen	- -	- -
Next Screen (E6)	Next Screen	- -	- -
HELP	Help	- -	- -
DO	Do	- -	- -
F6, F7, ..., F20	F6, F7, ..., F20	- -	- -

Some definable keys are enabled for definition all the time. Others, including KP0 through KP9, PERIOD, COMMA, and MINUS, must be

enabled for definition purposes. You must enter either the SET TERMINAL /APPLICATION or SET TERMINAL/NONUMERIC command before using these keys.

On LK201 keyboards, you cannot define the UP and DOWN arrow keys or function keys F1 through F5. The LEFT and RIGHT arrow keys and the F6 through F14 keys are reserved for command line editing. You must enter the SET TERMINAL/NOLINE_EDITING command before defining these keys. You can also press CTRL/V to enable keys F7 through F14. Note that CTRL/V will not enable the F6 key.

equivalence-string

Specifies the character string to be processed when you press the key. Enclose the string in quotation marks to preserve spaces and lowercase characters.

DESCRIPTION

The DEFINE/KEY command enables you to assign definitions to the peripheral keys on certain terminals. The terminals include VT52s, the VT100 series, and terminals with LK201 keyboards.

To define keys on the numeric keypads of these terminals, you must first enter the SET TERMINAL/APPLICATION or SET TERMINAL /NONUMERIC command. When your terminal has this setting, the system interprets the keystrokes from keypad keys differently. For example, with SET TERMINAL/NONUMERIC in effect, pressing the 1 key on the keypad does not send the character "1" to the system.

The equivalence string definition can contain different types of information. Definitions often consist of DCL commands. For example, you can assign SHOW TIME to the zero key. When you press 0, the system displays the current date and time. Other definitions can consist of text strings to be appended to command lines. When you define a key to insert a text string, use the /NOTERMINATE qualifier so that you can continue typing more data after the string has been inserted.

In most instances you will want to use the echo feature. The default setting is /ECHO. With /ECHO set, the key definition is displayed on the screen each time you press the key.

You can use the /STATE qualifier to increase the number of key definitions available on your terminal. The same key can be assigned any number of definitions, as long as each definition is associated with a different state. State names can contain any alphanumeric characters, dollar signs, and underscores. Be sure to create a state name that is easy to remember and type and, if possible, one that might remind you of the types of definitions you created for that state. For example, you can create a state called SETSHOW. The key definitions for this state might all refer to various DCL SET and SHOW commands. If you are used to the EDT Editor, you might define a state as GOLD. Then, using the /IF_STATE qualifier, you can assign different definitions to keys used in combination with a key defined as GOLD.

The SET KEY command changes the keypad state. Use the SHOW KEY command to display key definitions and states.

QUALIFIERS

/ECHO (default)
/NOECHO

Displays the equivalence string on your screen after the key has been pressed. You cannot use /NOECHO with the /NOTERMINATE qualifier.

/ERASE
/NOERASE (default)

Determines whether the current line is erased before the key translation is inserted.

/IF_STATE=(state-name,...)
/NOIF_STATE

Specifies a list of one or more states, one of which must be in effect for the key definition to work. The /NOIF_STATE has the same meaning as /IF_STATE=current_state. The state name is an alphanumeric string. States are established with the /SET_STATE qualifier or the SET KEY command. If you specify only one state name, you can omit the parentheses. By including several state names, you can define a key to have the same function in all the specified states.

/LOCK_STATE
/NOLOCK_STATE (default)

Specifies that the state set by the /SET_STATE qualifier remain in effect until explicitly changed. (By default, the /SET_STATE qualifier is in effect only for the next definable key you press or the next read-terminating character that you type.) Can only be specified with the /SET_STATE qualifier.

/LOG (default)
/NOLOG

Displays a message indicating that the key definition has been successfully created.

/SET_STATE=state-name
/NOSET_STATE (default)

Causes the specified state-name to be set when the key is pressed. (By default, the current locked state is reset when the key is pressed.) If you have not included this qualifier with a key definition, you can use the SET KEY command to change the current state. The state name can be any alphanumeric string; specify the state as a character string enclosed in quotation marks (").

/TERMINATE
/NOTERMINATE (default)

Specifies whether the current equivalence string is to be processed immediately when the key is pressed (equivalent to entering the string and pressing RETURN). By default, you can press other keys before the definition is processed. This allows you to create key definitions that insert text into command lines, after prompts, or into other text that you are entering.

EXAMPLES

1
```
$ DEFINE/KEY PF3 "SHOW TIME" /TERMINATE
%DCL-I-DEFKEY, DEFAULT key PF3 has been defined
$ SHOW TIME
  15-APR-1988 14:43:59
```

In this example, the DEFINE/KEY command defines the PF3 key on the keypad to perform the SHOW TIME command. DEFAULT refers to the default state.

2
```
$ DEFINE/KEY PF1 "SHOW " /SET_STATE=GOLD/NOTERMINATE/ECHO
%DCL-I-DEFKEY, DEFAULT key PF1 has been defined
$ DEFINE/KEY PF1 " DEFAULT" /TERMINATE/IF_STATE=GOLD/ECHO
%DCL-I-DEFKEY, GOLD key PF1 has been defined
$ SHOW DEFAULT
DISK1:[JOHN.TEST]
```

In this example, the first DEFINE/KEY command defines the PF1 key to be the string SHOW. The state is set to GOLD for the subsequent key. The /NOTERMINATE qualifier instructs the system not to process the string when the key is pressed. The second DEFINE/KEY command defines the use of the PF1 key when the keypad is in the GOLD state. When the keypad is in the GOLD state, pressing PF1 causes the current read to be terminated.

If you press the PF1 key twice, the system displays and processes the SHOW DEFAULT command.

The word *DEFAULT* in the second line of the example indicates that the PF1 key has been defined in the default state. Note the space before the word *DEFAULT* in the second DEFINE/KEY command. If the space is omitted, the system fails to recognize DEFAULT as the keyword for the SHOW command.

3
```
$ SET KEY/STATE=ONE
%DCL-I-SETKEY, keypad state has been set to ONE
$ DEFINE/KEY PF1 "ONE"
%DCL-I-DEFKEY, ONE key PF1 has been defined
$ DEFINE/KEY/IF_STATE=ONE PF1 "ONE"
%DCL-I-DEFKEY, ONE key PF1 has been defined
```

The previous two examples define the PF1 key to be "ONE" for state ONE.

The second example shows the prefered method for defining keys. This method eliminates the possibility of error by specifying the state in the same command as the key definition.

DELETE

Deletes one or more files from a mass storage disk volume.

FORMAT

DELETE *file-spec[,...]*

PARAMETER

file-spec[,...]

Specifies the names of one or more files to be deleted from a mass storage disk volume. The first file specification must contain an explicit or default directory specification plus an explicit file name, file type, and version number. Subsequent file specifications need contain only a version number; the defaults will come from the preceding specification. Wildcard characters can be used in any of the file specification fields.

If you omit the directory specification or device name, the current default device and directory are assumed.

If the file specification contains a null version number (a semicolon followed by no file version number), a version number of 0, or one or more spaces in the version number, the latest version of the file is deleted.

To delete more than one file, separate the file specifications with commas or plus signs.

QUALIFIERS

/BACKUP

Modifies the time value specified with the /BEFORE or /SINCE qualifier. /BACKUP selects files according to the dates of their most recent backups. This qualifier is incompatible with the other qualifiers that also allow you to select files according to time attributes: /CREATED, /EXPIRED, and /MODIFIED. If you specify none of these four time qualifiers, the default is /CREATED.

/BEFORE[=time]

Selects only those files dated prior to the specified time. You can specify time as an absolute time, as a combination of absolute and delta times, or as one of the following keywords: TODAY (default), TOMORROW, or YESTERDAY. Specify one of the following qualifiers with /BEFORE to indicate the time attribute to be used as the basis for selection: /BACKUP, /CREATED (default), /EXPIRED, or /MODIFIED.

See Section 1.4 of the *VMS DCL Concepts Manual* for complete information on specifying time values.

/BY_OWNER[=uic]

Selects only those files whose owner user identification code (UIC) matches the specified owner UIC. The default UIC is that of the current process.

Specify the UIC using standard UIC format as described in Section 8.1 of the *VMS DCL Concepts Manual.*

/CONFIRM
/NOCONFIRM (default)

Controls whether a request is issued before each DELETE operation to confirm that the operation should be performed on that file. The following responses are valid:

YES	NO	QUIT
TRUE	FALSE	CTRL/Z
1	0	ALL
	RET	

You can use any combination of upper- and lowercase letters for word responses. Word responses can be abbreviated to one or more letters (for example, T, TR, or TRU for TRUE), but these abbreviations must be unique. Affirmative answers are YES, TRUE, and 1. Negative answers are NO, FALSE, 0, and the RETURN key. QUIT or CTRL/Z indicates that you want to stop processing the command at that point. When you respond with ALL, the command continues to process, but no further prompts are given. If you type a response other than one of those in the list, DCL issues an error message and redisplays the prompt.

/CREATED (default)

Modifies the time value specified with the /BEFORE or /SINCE qualifier. /CREATED selects files based on their dates of creation. This qualifier is incompatible with the other qualifiers that also allow you to select files according to time attributes: /BACKUP, /EXPIRED, and /MODIFIED. If you specify none of these four time qualifiers, the default is /CREATED.

/ERASE
/NOERASE (default)

When you delete a file, the area in which the file was stored is returned to the system for future use. The data that was stored in that location still exists in the system until new data is written over it. When you specify the /ERASE qualifier, the storage location is overwritten with a system specified pattern so that the data no longer exists.

/EXCLUDE=(file-spec[,...])

Excludes the specified files from the DELETE operation. You can include a directory but not a device in the file specification. Wildcard characters are allowed in the file specification. However, you cannot use relative version numbers to exclude a specific version. If you provide only one file specification, you can omit the parentheses.

/EXPIRED

Modifies the time value specified with the /BEFORE or /SINCE qualifier. /EXPIRED selects files according to their expiration dates. (The expiration date is set with the SET FILE/EXPIRATION_DATE command.) The /EXPIRED qualifier is incompatible with the other qualifiers that also allow you to select files according to time attributes: /BACKUP, /CREATED, and /MODIFIED. If you specify none of these four time qualifiers, the default is /CREATED.

/LOG
/NOLOG (default)

Controls whether the DELETE command displays the file specification of each file after its deletion.

/MODIFIED

Modifies the time value specified with the /BEFORE or /SINCE qualifier. /MODIFIED selects files according to the dates on which they were last modified. This qualifier is incompatible with the other qualifiers that also allow you to select files according to time attributes: /BACKUP, /CREATED, and /EXPIRED. If you specify none of these four time modifiers, the default is /CREATED.

/SINCE[=time]

Selects only those files dated after the specified time. You can specify time as an absolute time, a combination of absolute and delta times, or as one of the following keywords: TODAY (default), TOMORROW, or YESTERDAY. Specify one of the following qualifiers with /BEFORE to indicate the time attribute to be used as the basis for selection: /BACKUP, /CREATED (default), /EXPIRED, or /MODIFIED.

See Section 1.4 of the *VMS DCL Concepts Manual* for complete information on specifying time values.

EXAMPLES

1
```
$ DELETE   COMMON.SUM;2
```

The DELETE command deletes the file COMMON.SUM;2 from the current default disk and directory.

2
```
$ DELETE *.OLD;*
```

The DELETE command deletes all versions of files with file type OLD from the default disk directory.

3
```
$ DELETE  ALPHA.TXT;*, BETA;*, GAMMA;*
```

The DELETE command deletes all versions of the files ALPHA.TXT, BETA.TXT, and GAMMA.TXT. The command uses the file type of the first input file as a temporary default. Note, however, that some form of version number (here specified as wildcards) must be included in each file specification.

4
```
$ DELETE /BEFORE=15-APR/LOG *.DAT;*
%DELETE-I-FILDEL, DISK2:[MALCOLM]ASSIGN.DAT;1 deleted (5 block)
%DELETE-I-FILDEL, DISK2:[MALCOLM]BATCHAVE.DAT;3 deleted (4 blocks)
%DELETE-I-FILDEL, DISK2:[MALCOLM]BATCHAVE.DAT;2 deleted (4 blocks)
%DELETE-I-FILDEL, DISK2:[MALCOLM]BATCHAVE.DAT;1 deleted (4 blocks)
%DELETE-I-FILDEL, DISK2:[MALCOLM]CANCEL.DAT;1 deleted (2 blocks)
%DELETE-I-FILDEL, DISK2:[MALCOLM]DEFINE.DAT;1 deleted (3 blocks)
%DELETE-I-FILDEL, DISK2:[MALCOLM]EXIT.DAT;1 deleted (1 block)
%DELETE-I-TOTAL, 7 files deleted (23 blocks)
```

The DELETE command deletes all versions of all files with file type DAT that were either created or updated before April 15 of this year. The /LOG

qualifier not only displays the name of each file deleted, but also the total number of files deleted.

5
```
$ DELETE A.B;
```

The DELETE command deletes the file A.B with the highest version number.

6
```
$ DELETE/CONFIRM/SINCE=TODAY [MALCOLM.TESTFILES]*.OBJ;*
DISK0:[MALCOLM.TESTFILES]AVERAG.OBJ;1, delete? [N]:Y
DISK0:[MALCOLM.TESTFILES]SCANLINE.OBJ;4, delete? [N]:N
DISK0:[MALCOLM.TESTFILES]SCANLINE.OBJ;3, delete? [N]:N
DISK0:[MALCOLM.TESTFILES]SCANLINE.OBJ;2, delete? [N]:N
DISK0:[MALCOLM.TESTFILES]WEATHER.OBJ;3, delete? [N]:Y
```

The DELETE command examines all versions of files with file type OBJ in the subdirectory [MALCOLM.TESTFILES], and locates those that were created or modified today. Before deleting each file, it requests confirmation that the file should be deleted. The default response—N—is given in square brackets.

7
```
$ DIRECTORY [.SUBTEST]
%DIRECT-W-NOFILES, no files found
$ SET PROTECTION SUBTEST.DIR/PROTECTION=OWNER:D
$ DELETE SUBTEST.DIR;1
```

Before the directory file SUBTEST.DIR is deleted, the DIRECTORY command is used to verify that there are no files cataloged in the directory. The SET PROTECTION command redefines the protection for the directory file so that it can be deleted; then the DELETE command deletes it.

8
```
$ DELETE DALLAS"THOMAS SECRET"::DISK0:[000,000]DECODE.LIS;1
```

This DELETE command deletes the file DECODE.LIS;1 from the directory [000,000] on device DISK0 at remote node DALLAS. The user name and password follow the remote node name.

9
```
$ DELETE QUEBEC::"DISK1:DEAL.BIG"
$ DELETE QUEBEC::DISK1:DEAL.BIG;
```

Either of these DELETE commands can be used to delete the file DEAL.BIG on device ZZZ1 at remote node QUEBEC. Note that the DELETE command requires an explicit version number in a file specification, but the file to be deleted is on a remote node whose file syntax does not recognize version numbers. (QUEBEC is an RT-11 node.) Therefore, the file specification must either be enclosed in quotation marks or entered with a null version number (that is, a trailing semicolon).

DELETE/CHARACTERISTIC

Deletes the definition of a queue characteristic previously established with the DEFINE/CHARACTERISTIC command. The full command, DELETE /CHARACTERISTIC, is required when deleting definitions.

Requires OPER privilege.

FORMAT

DELETE/CHARACTERISTIC *characteristic-name*

PARAMETER

characteristic-name
Specifies the name of the characteristic.

DESCRIPTION

The DELETE/CHARACTERISTIC command deletes a characteristic from the system characteristic table.

To change the number of an existing characteristic, you can use the DEFINE /CHARACTERISTIC command. It is not necessary to delete the characteristic before changing it.

EXAMPLE

```
$ DEFINE/CHARACTERISTIC BLUE 7
   .
   .
   .
$ DELETE/CHARACTERISTIC BLUE
$ DEFINE/CHARACTERISTIC BLUE_INK 7
```

The DEFINE/CHARACTERISTIC command in this example establishes the characteristic BLUE, with number 7, to mean blue ink ribbons for printers. To change the name of the characteristic, enter the DELETE/CHARACTERISTIC command. Then enter another DEFINE/CHARACTERISTIC command to rename the characteristic to BLUE_INK, using the characteristic number 7.

DELETE/ENTRY

Deletes one or more print or batch jobs from a queue. The jobs can be in progress or waiting in the queue. The full command, DELETE/ENTRY, is required to delete jobs from a queue.

Requires OPER privilege, EXECUTE access to the queue, or DELETE access to the job.

FORMAT

DELETE/ENTRY=*(job-number[,...]) [queue-name:]*

PARAMETERS

job-number[,...]

Specifies the job number of a job to be deleted from the queue. The DELETE /ENTRY command requires at least one job-number parameter, specifying one or more jobs to be deleted from a single printer or batch queue. If you specify only one job number, you can omit the parentheses.

queue-name[:]

Specifies the name of the queue where the jobs are located. The queue name can refer either to the queue to which the job was submitted or to the queue where the job is executing. The queue-name parameter is optional syntax. However, when it is specified, VMS uses queue-name to verify an entry in the specific queue before deleting the entry.

DESCRIPTION

The DELETE/ENTRY command deletes one or more jobs from a queue. If you specify more than one entry number with a DELETE/ENTRY command, all the jobs must be located in the same queue.

You can delete jobs that are currently executing, as well as jobs that are in other states. For example, DELETE/ENTRY can stop a job that is currently printing.

EXAMPLES

1
```
$ PRINT/HOLD   ALPHA.TXT
Job ALPHA (queue SYS$PRINT, entry 110) holding
   .
   .
   .
$ DELETE/ENTRY=110  SYS$PRINT
```

The PRINT command in this example queues a copy of the file ALPHA.TXT in a HOLD status, to defer its printing until a SET QUEUE/ENTRY/RELEASE command is entered. The system displays the job name, entry number, name of the queue in which the job was entered, and the status. Later, the DELETE/ENTRY command requests that the entry be deleted from the queue SYS$PRINT.

DELETE/ENTRY

2

```
$ SUBMIT/HOLD/PARAMETERS=SCANLINE  DOFOR
  Job DOFOR (queue SYS$BATCH, entry 203) holding
$ SUBMIT/AFTER=18:00  WEATHER
  Job WEATHER (queue SYS$BATCH, entry 210) holding until 15_APR-1988 18:00
   .
   .
   .
$ DELETE/ENTRY=(203,210)   SYS$BATCH
```

The SUBMIT commands in this example queue the command procedures DOFOR.COM and WEATHER.COM for processing as batch jobs. DOFOR.COM is queued in a HOLD status and cannot execute until you enter a SET QUEUE/ENTRY/RELEASE command. WEATHER.COM is queued for execution after 6:00 P.M. Later, the DELETE/ENTRY command requests that both these entries be deleted from the queue SYS$BATCH.

3

```
$ PRINT CHAPTER8.MEM
Job CHAPTER8 (queue SYS$PRINT, entry 25) pending on queue LPA0
   .
   .
   .
$ SHOW QUEUE SYS$PRINT
Printer queue SYS$PRINT, on LPA0:
Jobname         Username        Entry   Blocks  Status
-------         --------        -----   ------  ------
CHAPTER7        SMITH              24      274  Pending
CHAPTER8        SMITH              25      976  Pending
$ DELETE/ENTRY=25 SYS$PRINT
```

The PRINT command in this example submits the file CHAPTER8.MEM to the generic printer queue SYS$PRINT. Later, user Smith needs to edit the file again before printing it. Using the SHOW QUEUE command, Smith verifies that the job is still pending and that the entry number for the job is 25. Smith then enters the DELETE/ENTRY command to delete the job from the queue.

DELETE/FORM

Deletes a form type for a printer or a terminal queue previously established with the DEFINE/FORM command. When you delete a form definition, you must ensure that no outstanding references to the form exist in queues that have been mounted with the form or by jobs requesting that form. The /FORM qualifier is required.

Requires OPER privilege.

FORMAT

DELETE/FORM *form-name*

PARAMETER

form-name
Specifies the name that was assigned to the form by a DEFINE/FORM command.

DESCRIPTION

The DELETE/FORM command deletes a form definition from the system forms table. When you delete a form, there can be no outstanding references to the form either in queues that have been mounted with the form or by jobs requesting that form. Use the SHOW QUEUE/FULL qualifier to locate all references to the form.

To change the number or attributes of an existing form, use the DEFINE /FORM command. It is not necessary to delete the form before changing it.

EXAMPLES

1

```
$ DELETE/FORM CENTER
```

The DELETE/FORM command in this example deletes the form named CENTER.

2

```
$ DEFINE/FORM /DESCRIPTION="letter size continuous form paper" CFLET 7
.
.
.
$ DELETE/FORM CFLET
$ DEFINE/FORM /DESCRIPTION="letter size continuous form paper" LETTER_CONT 7
```

The DEFINE/FORM command in this example establishes the form CFLET with number 7, to mean 8.5 by 11 inch continuous form feed paper. To change the name of the form, delete the form named CFLET and define a new one named LETTER_CONT.

DELETE/INTRUSION_RECORD

Removes an entry from the break-in database.

Requires CMKRNL and SECURITY privileges.

FORMAT

DELETE/INTRUSION_RECORD *source*

PARAMETER

source
Source field of the entry to be removed from the break-in database.

DESCRIPTION

Use the DELETE/INTRUSION_RECORD command to remove an entry from the break-in database. For example, if the user Hammer repeatedly attempted to log in on terminal TTA24 with an expired password, the SHOW INTRUSION command would display the following entry:

```
Intrusion  Type      Count    Expiration   Source

TERM_USER   INTRUDER    9      10:29:39.16   TTA24:HAMMER
```

The terminal is locked out of the system because the login failure limit has been reached. When Hammer approaches you and you identify the problem as an expired password, you can then use the DELETE/INTRUSION command to remove the record from the break-in database.

EXAMPLES

1 `$ DELETE/INTRUSION_RECORD TTC2:`

In this example, the DELETE/INTRUSION_RECORD command removes all intrusion records generated by break-in attempts on TTC2. No username is specified because none of the login failures occurred for valid users.

2 `$ DELETE/INTRUSION_RECORD GALAXY::HAMMER`

This command removes all intrusion entries generated from node GALAXY for user HAMMER.

DELETE/KEY

Deletes key definitions that have been established by the DEFINE/KEY command. The /KEY qualifier is required.

FORMAT

DELETE/KEY *[key-name]*

PARAMETER

key-name

Specifies the name of the key to be deleted. Incompatible with the /ALL qualifier.

QUALIFIERS

/ALL

Deletes all key definitions in the specified state; the default is the current state. If you use the /ALL qualifier, do not specify a key name. Use the /STATE qualifier to specify one or more states.

/LOG (default)
/NOLOG

Controls whether messages are displayed indicating that the specified key definitions have been deleted.

/STATE=(state-name[,...])
/NOSTATE (default)

Specifies the name of the state for which the specified key definition is to be deleted. The default state is the current state. If you specify only one state name, you can omit the parentheses. State names can be any appropriate alphanumeric string.

EXAMPLES

1

```
$ DEFINE/KEY PF3 "SHOW TIME" /TERMINATE
%DCL-I-DEFKEY, DEFAULT key PF3 has been defined
$ [PF3]
$ SHOW TIME
 15-APR-1988 14:43:59
    .
    .
    .
$ DELETE/KEY PF3
%/DCL-I-DELKEY, DEFAULT key PF3 has been deleted
$ [PF3]
$
```

In this example, the DEFINE/KEY command defines the PF3 key on the keypad as SHOW TIME. To undefine the PF3 key, use the DELETE/KEY command. When the user presses PF3, only the system prompt is displayed.

```
2   $ DELETE/KEY/ALL
    %DCL-I-DELKEY, DEFAULT key PF1 has been deleted
    %DCL-I-DELKEY, DEFAULT key PF2 has been deleted
    %DCL-I-DELKEY, DEFAULT key PF3 has been deleted
    %DCL-I-DELKEY, DEFAULT key PF4 has been deleted
    $
```

In this example, the user defined keys PF1 through PF4 in the default state. The DELETE/KEY command deletes all key definitions in the current state, which is the default state.

DELETE/QUEUE

Deletes a print or batch queue and all the jobs in the queue. The specified queue must be stopped first.

Requires OPER privilege.

FORMAT

DELETE/QUEUE *queue-name[:]*

PARAMETER

queue-name[:]
Specifies the name of the queue to be deleted.

DESCRIPTION

The DELETE/QUEUE command takes effect only if the specified queue has been stopped. To delete a queue, do the following:

1. Be sure that there are no outstanding references to the queue. Queue references are made with the /PRINTER=queue-name and /GENERIC=queue-name qualifiers of the INITIALIZE/QUEUE, SET QUEUE, and START/QUEUE commands. The ASSIGN/QUEUE command also makes queue references.
2. After you have determined that there are no outstanding references to the queue, stop the queue with the STOP/QUEUE/NEXT command.
3. Wait for any current jobs to complete.
4. Enter the DELETE/QUEUE command. Note that any pending jobs in the queue are deleted when the queue is deleted.

EXAMPLE

```
$ INITIALIZE/QUEUE/DEFAULT=FLAG/START LPA0
   .
   .
   .
$ STOP/QUEUE/NEXT LPA0
$ DELETE/QUEUE LPA0
```

In this example, the first command initializes and starts the printer queue LPA0. The STOP/QUEUE/NEXT command stops the queue. The DELETE /QUEUE command deletes the queue.

DELETE/SYMBOL

Deletes one or all symbol definitions from a local or global symbol table. The /SYMBOL qualifier is required.

FORMAT

DELETE/SYMBOL *[symbol-name]*

PARAMETER

symbol-name

Specifies the name of the symbol to be deleted. A name is required unless the /ALL qualifier is specified. The symbol-name parameter is incompatible with the /ALL qualifier. Symbol names can have from 1 to 255 characters. By default, the DELETE/SYMBOL command assumes that the symbol is in the local symbol table for the current command procedure.

DESCRIPTION

The DELETE/SYMBOL command deletes a symbol definition from a symbol table. If you do not specify either the global or local symbol table, the symbol is deleted from the local table. If you specify both /GLOBAL and /LOCAL, only the last specified qualifier is accepted. The /SYMBOL qualifier must always immediately follow the DELETE command name.

QUALIFIERS

/ALL

Deletes all symbols from the specified table. If you do not specify either /LOCAL or /GLOBAL, all symbols defined at the current command level are deleted. The /ALL qualifier is incompatible with the symbol-name parameter.

/GLOBAL

Deletes the symbol from the global symbol table of the current process.

/LOCAL (default)

Deletes the symbol from the local symbol table of the current process.

/LOG
/NOLOG (default)

Controls whether an informational message listing each symbol being deleted is displayed.

EXAMPLES

1
```
$ DELETE/SYMBOL/ALL
```

In this example, the DELETE/SYMBOL command deletes all symbol definitions at the current command level.

2
```
$ DELETE/SYMBOL/LOG FOO
%DCL-I-DELSYM, LOCAL symbol FOO has been deleted
```

In this example, the DELETE/SYMBOL command deletes the symbol FOO from the local symbol table for the current process. In addition, the /LOG qualifier causes an informational message, listing the symbol being deleted, to be displayed.

3
```
$ DELETE/SYMBOL/GLOBAL PDEL
```

In this example, the DELETE/SYMBOL command deletes the symbol named PDEL from the global symbol table for the current process.

DEPOSIT

Replaces the contents of the specified locations in virtual memory and displays the new contents. If the specified address can be read but not written by the current access mode, the original contents are displayed; if the specified address can be neither read nor written, asterisks are displayed in the data field. The DEPOSIT command maintains a pointer at that location (at the byte following the last byte modified).

The DEPOSIT command, together with the EXAMINE command, aids in debugging programs interactively. The DCL DEPOSIT command is similar to the DEPOSIT command of the VMS Symbolic Debugger.

Requires user mode read (R) and write (W) access to the virtual memory location whose contents you wish to change.

FORMAT

DEPOSIT *location=data[,...]*

PARAMETERS

location

Specifies the starting virtual address or range of virtual addresses (where the second address is larger than the first) whose contents are to be changed. A location can be any valid integer expression containing an integer value, a symbol name, a lexical function, or a combination of these entities. Radix qualifiers determine the radix in which the address is interpreted; hexadecimal is the initial default radix. Symbol names are always interpreted in the radix in which they were defined. The radix operators %X, %D, or %O can precede the location. A hexadecimal value must begin with a number (or be preceded by %X).

The specified location must be within the virtual address space of the image currently running in the process.

The DEPOSIT and EXAMINE commands maintain a pointer to a current memory location. The DEPOSIT command sets this pointer to the byte following the last byte modified; you can refer to this pointer by using a period (.) in subsequent EXAMINE and DEPOSIT commands. If the DEPOSIT command cannot deposit the specified data, the pointer does not change. The EXAMINE command does not change the value of the pointer.

data[,...]

Specifies the data to be deposited into the specified locations. By default, the data is assumed to be in hexadecimal format; it is then converted to binary format and is written into the specified location.

If you specify a list, separate the items with commas; the DEPOSIT command writes the data in consecutive locations, beginning with the address specified.

When non-ASCII data is deposited, you can specify each item of data using any valid integer expression.

When ASCII data is deposited, only one item of data is allowed. All characters to the right of the equal sign are considered to be part of a single string. The characters are converted to uppercase, and all spaces are compressed.

DESCRIPTION

When the DEPOSIT command completes, it displays both the virtual memory address into which data is deposited and the new contents of the location, as follows:

```
address:  contents
```

If the specified address can be read but not written by the current access mode, the DEPOSIT command displays the original contents of the location. If the specified address can be neither read nor written, the DEPOSIT command displays asterisks in the data field.

If you specify a list of numeric values, some but not all of the values may be successfully deposited before an access violation occurs. If an access violation occurs while ASCII data is being deposited, nothing is deposited.

Radix Qualifiers: The radix default for a DEPOSIT or EXAMINE command determines how the command interpreter interprets numeric literals. The initial default radix is hexadecimal; all numeric literals in the command line are assumed to be hexadecimal values. If a radix qualifier modifies the command, that radix becomes the default for subsequent EXAMINE and DEPOSIT commands, until another qualifier overrides it. For example:

```
$ DEPOSIT/DECIMAL 900=256
00000384:  256
```

The DEPOSIT command interprets both the location 900 and the value 256 as decimal. All subsequent DEPOSIT and EXAMINE commands assume that numbers you enter for addresses and data are decimal. Note that the DEPOSIT command always displays the address location in hexadecimal.

Symbol values defined by = (Assignment Statement) commands are always interpreted in the radix in which they were defined.

Note that hexadecimal values entered as deposit locations or as data to be deposited must begin with a numeric character (0 through 9). Otherwise, the command interpreter assumes that you have entered a symbol name and attempts symbol substitution.

You can use the radix operators %X, %D, or %O to override the current default when you enter the DEPOSIT command. For example:

```
$ DEPOSIT/DECIMAL %X900=10
```

This command deposits the decimal value 10 in the location specified as hexadecimal 900.

Length Qualifiers: The initial default length unit for the DEPOSIT command is a longword. If a list of data values is specified, the data is deposited into consecutive longwords beginning at the specified location. If a length qualifier modifies the command, that length becomes the default for subsequent EXAMINE and DEPOSIT commands, until another qualifier overrides it. If you specify data values that are longer than the specified length, an error occurs.

Length qualifiers are ignored when ASCII values are deposited.

Restriction on Placement of Qualifiers: The DEPOSIT command analyzes expressions arithmetically. Therefore, qualifiers, which must be preceded by a slash (/), must appear immediately after the command name to be interpreted correctly.

DEPOSIT

QUALIFIERS

/ASCII
Indicates that the specified data is ASCII.

Only one data item is allowed; all characters to the right of the equal sign are considered to be part of a single string. Unless they are enclosed within quotation marks, characters are converted to uppercase and multiple spaces are compressed to a single space before the data is written in memory.

The DEPOSIT command converts the data to its binary equivalent before placing it in virtual memory. When you specify /ASCII, or when ASCII mode is the default, the location you specify is assumed to be hexadecimal.

/BYTE
Requests that data be deposited one byte at a time.

/DECIMAL
Indicates that the data is decimal. The DEPOSIT command converts the data to its binary equivalent before placing it in virtual memory.

/HEXADECIMAL
Indicates that the data is hexadecimal. The DEPOSIT command converts the data to its binary equivalent before placing it in virtual memory.

/LONGWORD
Requests that data be deposited a longword at a time.

/OCTAL
Indicates that the data is octal. The DEPOSIT command converts the data to its binary equivalent before placing it in virtual memory.

/WORD
Requests that the data be deposited one word at a time.

EXAMPLES

1
```
$ RUN MYPROG
   .
   .
   .
CTRL/Y
$ EXAMINE 2780
00002780:  1C50B344
$ DEPOSIT .=0
00002780:  00000000
$ CONTINUE
```

The RUN command executes the image MYPROG.EXE; subsequently, CTRL/Y interrupts the program. Assuming that the initial defaults of /HEXADECIMAL and /LONGWORD are in effect, the DEPOSIT command places a longword of zeros in virtual memory location 2780.

Because the EXAMINE command sets up a pointer to the current memory location, which in this case is virtual address 2780, you can refer to this location with "." in the DEPOSIT command.

The CONTINUE command resumes execution of the image.

```
2  $ DEPOSIT/ASCII   2C00=FILE: NAME: TYPE:
   00002C00:  FILE: NAME: TYPE:...
```

In this example, the DEPOSIT command deposits character data at hexadecimal location 2C00 and displays the contents of the location after modifying it. Because the current default length is a longword, the response from the DEPOSIT command displays full longwords. Trailing dots (ellipses) indicate that the remainder of the last longword of data contains information that was not modified by the DEPOSIT command.

```
3  $ EXAMINE 9C0         !  Look at Hex location 9C0
   000009C0:  8C037DB3
   $ DEPOSIT .=0         !  Deposit longword of 0
   000009C0:  00000000
   $ DEPOSIT/BYTE .=1    !  Put 1 byte at next location
   000009C4:  01
   $ DEPOSIT .+2=55      !  Deposit 55 next
   000009C7:  55
   $ DEPOSIT/LONG .=0C,0D,0E !  Deposit longwords
   000009C8:  0000000C 0000000D 0000000E
```

The sequence of DEPOSIT commands in the above example illustrates how the DEPOSIT command changes the current position pointer. Note that after you specify /BYTE, all data is deposited and displayed in bytes, until the /LONGWORD qualifier restores the system default.

```
4  $ BASE=%X200          !  Define a base address
   $ LIST=BASE+%X40      !  Define offset from base
   $ DEPOSIT/DECIMAL LIST=1,22,333,4444
   00000240:  00000001 00000022 00000333 00004444
   $ EXAMINE/HEX LIST:LIST+0C !  Display results in hex
   00000240:  00000001 00000016 0000014D 0000115C
```

The assignment statements define a base address in hexadecimal and a label at a hexadecimal offset from the base address. The DEPOSIT command reads the list of values and deposits each value into a longword, beginning at the specified location. The EXAMINE command requests a hexadecimal display of these values.

DIFFERENCES

Compares the contents of two disk files and displays a listing of the records that do not match.

FORMAT

DIFFERENCES *input1-file-spec [input2-file-spec]*

PARAMETERS

input1-file-spec

Specifies the first file to be compared. The file specification must include a file name and a file type. Wildcard characters are not allowed.

input2-file-spec

Specifies the second file to be compared. Unspecified fields default to the corresponding fields in input1-file-spec. Wildcard characters are not allowed.

If you do not specify a secondary input file, the DIFFERENCES command uses the next lower version of the primary input file.

DESCRIPTION

Use the DIFFERENCES command to find out whether two files are identical and, if not, how they differ. The DIFFERENCES command compares the two specified files on a record-by-record basis and produces an output file that lists the differences, if any.

The qualifiers for the DIFFERENCES command can be categorized according to their functions, as follows:

- Qualifiers that request DIFFERENCES to ignore data in each record:

  /COMMENT_DELIMITERS
  /IGNORE

  These qualifiers allow you to define characters that denote comments or to designate characters or classes of characters to ignore when comparing files. For example, you can have DIFFERENCES ignore extra blank lines or extra spaces within lines.

  By default, DIFFERENCES compares every character in each record.

- Qualifiers that control the format of the information contained in the list of differences:

  /CHANGE_BAR
  /IGNORE
  /MERGED
  /MODE
  /PARALLEL
  /SEPARATED
  /SLP
  /WIDTH

  By default, DIFFERENCES merges the differences it finds in the files being compared. It lists each record in the file that has no match in the other input file and then lists the next record that it finds that does have a match.

By default, DIFFERENCES also supplies a line number with each listed record, and it lists the records with all designated ignore characters deleted.

You can specify combinations of qualifiers to request an output listing that includes the comparison in more than one format. Note that SLP output is incompatible with all other types of output; PARALLEL output can be generated only in ASCII mode.

- Qualifiers that control the extent of the comparison:

  /MATCH
  /MAXIMUM_DIFFERENCES
  /WINDOW

  By default, DIFFERENCES reads every record in the master input file and looks for a matching record in the revision input file. A search for a match between the two input files continues until either a match is found or the ends of the two files are reached. Sections of the two files are considered a match only if three sequential records are found to be identical in each file.

By default, DIFFERENCES output is written to the current SYS$OUTPUT device. Use the /OUTPUT qualifier to request DIFFERENCES to write the output to an alternate file or device.

DIFFERENCES terminates with an exit status. The following severity levels indicate the result of the comparison:

SUCCESS	Files are identical.
INFORMATIONAL	Files are different.
WARNING	User-specified maximum number of differences has been exceeded.
ERROR	Insufficient virtual memory to complete comparison.

All severity levels other than SUCCESS indicate that the two input files are different.

QUALIFIERS

/CHANGE_BAR[=([change-char][,[NO]NUMBER])]

Marks with the specified character in the left margin each line in the input1 file that differs from the corresponding line in the input2 file. If you do not specify a change bar character, the default is an exclamation point (!) for ASCII output. If you specify hexadecimal or octal output (see /MODE qualifier), the change bar character is ignored and differences are marked by a "***CHANGE***" string in the record header. The keyword NONUMBER suppresses line numbers in the listing. If neither the NUMBER nor NONUMBER keyword is specified, the default is controlled by the /[NO]NUMBER command qualifier. If only one option is specified, the parentheses can be omitted. To specify both a change bar character and either NUMBER or NONUMBER, separate the options with a comma and enclose the list in parentheses, for example, /CHANGE_BAR=($,NUMBER).

/COMMENT_DELIMITER[=(character[,...])]

Ignores lines starting with a specified comment character. If the comment character is an exclamation point or semicolon, it can appear anywhere in the line and characters to the right of the character are ignored. If you specify just one character, you can omit the parentheses. Lowercase characters are

automatically converted to uppercase unless they are enclosed in quotation marks. Non-alphanumeric characters (such as ! and ,) must be enclosed in quotation marks. You can specify up to 32 comment characters by typing the character itself or one of the following keywords. (Keywords can be abbreviated provided that the resultant keyword is not ambiguous and has at least two characters; single letters are treated as delimiters.)

Keyword	Character
COLON	Colon (:)
COMMA	Comma (,)
EXCLAMATION	Exclamation point (!)
FORM_FEED	Form feed
LEFT	Left bracket ([)
RIGHT	Right bracket (])
SEMI_COLON	Semicolon (;)
SLASH	Slash (/)
SPACE	Space
TAB	Tab

The /COMMAND_DELIMITER qualifier is used with or without the /IGNORE=COMMENTS qualifier to indicate which comments are to be ignored.

If both the uppercase and lowercase forms of a letter are to be used as comment characters, the letter must be specified twice, once in uppercase and once in lowercase. If you do not include either a comment character or a keyword with the /COMMAND_DELIMITER qualifier, DIFFERENCES assumes a default comment character based on the file type. For some file types (COB and FOR), the default comment characters are considered valid delimeters only if they appear in the first column of a line. Multicharacter comment characters are not allowed.

The following characters are the default comment delimiters for files with the specified file types.

File type	Default Comment Character
B2S, B32, BAS, BLI	!
CBL, CMD	! and ;
COB	* or / in the first column
COM, COR	!
FOR	! anywhere and C, D, c, d in the first column
HLP	!
MAC, MAR	;
R32, REQ	!

/IGNORE=(keyword[,...])

Inhibits the comparison of the specified characters, strings, or records; also controls whether the comparison records are output to the listing file as edited records or exactly as they appeared in the input file. If you specify only one

keyword, you can omit the parentheses. The keyword parameter refers either to a character or a keyword. The first set of keywords determines what, if anything, is ignored during file comparison; the second set of keywords determines whether or not ignored characters are included in the output. The following keywords are valid options for the /IGNORE qualifier:

BLANK_LINES	Blank lines between data lines.
COMMENTS	Data following a comment character. (Use the /COMMENT_DELIMITER qualifier to designate one or more non-default comment delimiters.)
FORM_FEEDS	Form feed character.
HEADER[=n]	First *n* records of the file, beginning with a record whose first character is a form feed. The first record is not ignored if the only character it contains is a form feed. (N indicates the number of records and defaults to 2. A record with a single form feed is not counted.)
TRAILING_SPACES	Space and tab characters at the end of a data line.
SPACING	Extra blank spaces or tabs within data lines.
EDITED	Omits ignored characters from the output records.
EXACT	Includes ignored characters in the output records.
PRETTY	Formats output records.

Each data line is checked for COMMENTS, FORM_FEEDS, HEADER, and SPACING before it is tested for TRAILING_SPACES and then BLANK_LINES. Therefore, if you direct DIFFERENCES to ignore COMMENTS, TRAILING_SPACES, and BLANK_LINES, it ignores a record that contains several spaces or blank lines followed by a comment.

By default, the DIFFERENCES command compares every character in each file and reports all differences. Also, by default, DIFFERENCES lists records in the output file with all ignored characters deleted.

If you specify /PARALLEL, output records are always formatted. To format output records, specify the following characters:

Character	Formatted Output
Tab (CTRL/I)	1-8 spaces
RETURN (CTRL/M)	<CR>
Line feed (CTRL/J)	<LF>
Vertical tab (CTRL/K)	<VT>
Form feed (CTRL/L)	<FF>
Other nonprinting characters	. (period)

/MATCH=size

Specifies the number of records that should indicate matching data after a difference is found. By default, after DIFFERENCES finds unmatched records, it assumes that the files once again match after it finds three sequential records that match. Use the /MATCH qualifier to override the default match size of 3.

You can increase the /MATCH value if you feel that DIFFERENCES is incorrectly matching sections of the master and revision input files after it has detected a difference.

/MAXIMUM_DIFFERENCES=n

Terminates DIFFERENCES after a specified number of unmatched records (specified with the n parameter) is found.

The number of unmatched records is determined by finding the maximum number of difference records for each difference section and adding them together.

If DIFFERENCES reaches the maximum number of differences that you specify, it will output only those records that were detected before the maximum was reached. Also, it will output, at most, one listing format and return a warning message.

By default, there is no maximum number of differences. All records in the specified input files are compared.

/MERGED[=n]

Specifies that the output file contain a merged list of differences with the specified number of matched records listed after each group of unmatched records. The specified number (the value n) must be less than or equal to the number specified in the /MATCH qualifier. By default, DIFFERENCES produces a merged listing with one matched record listed after each set of unmatched records (that is, /MERGED=1). If neither /MERGED nor /SEPARATED nor /PARALLEL is specified, the resulting output is merged, with one matched record following each unmatched record.

Use the /MERGED qualifier to override the default value of n, or to include a merged listing with other types of output.

/MODE=(radix[,...])

Specifies the format of the output. You can request that the output be formatted in one or more radix modes by specifying the following keywords, which may be abbreviated: ASCII (default), HEXADECIMAL, or OCTAL. If you specify only one radix, you can omit the parentheses.

By default, DIFFERENCES writes the output file in ASCII. If you specify more than one radix, the output listing contains the file comparison in each specified radix. When you specify two or more radix modes, separate them with commas.

If you specify /PARALLEL or /SLP, /MODE is ignored for that listing form.

/NUMBER (default)
/NONUMBER

Includes line numbers in the listing of differences.

/OUTPUT[=file-spec]

Specifies an output file to receive the list of differences. By default, the output is written to the current SYS$OUTPUT device. If the file-spec parameter is not specified, the output is directed to the first input file with a file type of DIF. No wildcard characters are allowed.

When you specify /OUTPUT, you can control the defaults applied to the output file specification as described in Section 1.3 of the *VMS DCL Concepts Manual*. The default output file type is DIF.

/PARALLEL[=n]

Lists the records with differences side by side. The value n specifies the number of matched records to merge after each unmatched record; the value n must be a non-negative decimal number less than or equal to the number specified in /MATCH.

By default, DIFFERENCES does not list records after each list of unmatched records. Also by default, DIFFERENCES creates only a list of merged differences.

/SEPARATED[=(input1-file-spec[,input2-file-spec])]

Lists sequentially only the records from the specified file that contain differences. If no files are specified, a separate listing is generated for each file. If only one file is specified, you can omit the parentheses. To specify the input1-file-spec parameter, use either the first input file specified as the DIFFERENCES parameter or the keyword MASTER. To specify the input2-file-spec parameter, use either the second input file specified as the DIFFERENCES parameter or the keyword REVISION.

By default, DIFFERENCES creates only a merged list of differences.

/SLP

Requests that DIFFERENCES produce an output file suitable for input to the SLP editor. If you use the /SLP qualifier, you cannot specify any of the following output file qualifiers: /MERGED, /PARALLEL, /SEPARATED, or /CHANGE_BAR.

Use the output file produced by the SLP qualifier as input to SLP to update the master input file, that is, to make the master input file match the revision input file.

When you specify /SLP and you do not specify /OUTPUT, DIFFERENCES writes the output file to a file with the same file name as the master input file with the file type DIF.

/WIDTH=n

Specifies the width of the lines in the output file. The default is 132 characters. If output is written to the terminal, /WIDTH is ignored and the terminal line width is used.

Use the SET TERMINAL command to change the terminal line width.

/WINDOW=size

Searches the number of records specified (the value n) before a record is declared as unmatched. By default, DIFFERENCES searches to the ends of both input files before listing a record as unmatched.

The window size is the minimum size of a differences section that will cause DIFFERENCES to lose synchronization between the two input files.

EXAMPLES

1

```
$ DIFFERENCES EXAMPLE.TXT
************
File DISK1:[GEORGE.TEXT]EXAMPLE.TXT;2
     1   DEMONSTRATION
     2   OF V3.0 DIFFERENCES
     3   UTILITY
******
File DISK1:[GEORGE.TEXT]EXAMPLE.TXT;1
     1   DEMONSTRATION
     2   OF VMS DIFFERENCES
     3   UTILITY
************
Number of difference sections found: 1
Number of difference records found: 2
DIFFERENCES/MERGED=1-
       DISK1:[GEORGE.TEXT]EXAMPLE.TXT;2
       DISK1:[GEORGE.TEXT]EXAMPLE.TXT;1
```

In this example, the DIFFERENCES command compares the contents of the two most recent versions of the file EXAMPLE.TXT in the current default directory. DIFFERENCES compares every character in every record and displays the results at the terminal.

2

```
$ DIFFERENCES/PARALLEL/WIDTH=80/COMMENT_DELIMITER="V" EXAMPLE.TXT
--------------------------------------------------------------------------------
File DISK1:[GEORGE.TEXT]EXAMPLE.TXT;2  |  File DISK1:[GEORGE.TEXT]EXAMPLE.TXT;1
------------------ 1 ---------------------------------- 1 -----------------
DEMONSTRATION                        |  <LF>DEMONSTRATION
--------------------------------------------------------------------------------
Number of difference sections found: 1
Number of difference records found: 1
DIFFERENCES/IGNORE=(COMMENTS)/COMMENT_DELIMITER=("V")/WIDTH=80/PARALLEL-
       DISK1:[GEORGE.TEXT]EXAMPLE.TXT;2-
       DISK1:[GEORGE.TEXT]EXAMPLE.TXT;1
```

The DIFFERENCES command compares the same files as in Example 1, but ignores all comments following the first "V" encountered by DIFFERENCES. The command also specifies that an 80-column parallel list of differences be displayed.

```
3  $ DIFFERENCES/WIDTH=80/MODE=(HEX,ASCII) EXAMPLE.TXT/CHANGE_BAR
   ************
   File DISK1:[GEORGE.TEXT]EXAMPLE.TXT;2
       1 ! DEMONSTRATION
       2 ! OF V3.0 DIFFERENCES
       3   UTILITY
   ************
   ************
   File DISK1:[GEORGE.TEXT]EXAMPLE.TXT;2
   RECORD NUMBER 1 (00000001) LENGTH 14 (0000000E)  ***CHANGE***
        204E 4F495441 5254534E 4F4D4544 DEMONSTRATION .. 000000
   RECORD NUMBER 2 (00000002) LENGTH 19 (00000013)  ***CHANGE***
    4E455245 46464944 20302E33 5620464F OF V3.0 DIFFEREN 000000
                               534543 CES..............  000010
   RECORD NUMBER 3 (00000003) LENGTH 7 (00000007)
                         595449 4C495455 UTILITY.........  000000
   ************
   Number of difference sections found: 1
   Number of difference records found: 2
   DIFFERENCES /WIDTH=80/MODE=(HEX,ASCII)
          DISK1:[GEORGE.TEXT]EXAMPLE.TXT;2/CHANGE_BAR-
          DISK1:[GEORGE.TEXT]EXAMPLE.TXT;1
```

The DIFFERENCES command compares the same files as in Example 1, but lists the differences in both hexadecimal and ASCII formats. The command also specifies that default change bars be used in the output. The default change bar notation for the hexadecimal output is ***CHANGE***. For the ASCII output, the default change bar character is the exclamation point.

```
4  $ DIFFERENCES/OUTPUT BOSTON::DISK2:TEST.DAT OMAHA::DISK1:[PGM]TEST.DAT
```

The DIFFERENCES command compares two remote files and displays any differences found. The first file is TEST.DAT on remote node BOSTON. The second file is also named TEST.DAT on remote node OMAHA. The DIFFERENCES output is located in the file DISK1:[PGM]TEST.DIF.

DIRECTORY

Provides a list of files or information about a file or group of files.

Requires READ (R) access to the directories or sufficient privilege to override the protection to obtain information. Requires READ access to the files or sufficient privilege to override the protection to obtain information other than the file name.

FORMAT

DIRECTORY *[file-spec[,...]]*

PARAMETER

file-spec[,...]

Specifies one or more files to be listed. The syntax of a file specification determines which files will be listed, as follows:

- If you do not enter a file specification, the DIRECTORY command lists all versions of the files in the current default directory.
- If you specify only a device name, the DIRECTORY command uses your default directory specification.
- Whenever the file specification does not include a file name, file type and a version number, all versions of all files in the specified directory are listed.
- If a file specification contains a file name or a file type, or both, and no version number, the DIRECTORY command lists all versions.
- If a file specification contains only a file name, the DIRECTORY command lists all files in the current default directory with that file type, regardless of file type and version number.
- If a file specification contains only a file type, the DIRECTORY command lists all files in the current default directory with that file type, regardless of file name and version number.

Wildcard characters can be used in the directory specification, file name, file type, or version number fields of a file specification to list all files that satisfy the components you specify. Separate multiple file specifications with either commas or plus signs.

DESCRIPTION

The DIRECTORY command lists the files contained in a directory. When you use certain qualifiers with the command, additional information is displayed, along with the names of the files.

The output of the DIRECTORY command depends on certain formatting qualifiers and their defaults. These qualifiers are as follows: /COLUMNS, /DATE, /FULL, /OWNER, /PROTECTION, and /SIZE. However, the files that are listed always appear in alphabetical order, with the highest-numbered versions first.

In studying the qualifiers and the capabilities they offer, watch for qualifiers that work together and for qualifiers that override other qualifiers. For example, if you specify the /FULL format, the system cannot display all the information in more than one column. Thus, if you specify both /COLUMNS and /FULL, the number of columns you request is ignored.

QUALIFIERS

/ACL

Controls whether the access control list (ACL) is displayed for each file. By default, DIRECTORY does not display the ACL for each file. The /ACL qualifier overrides the /COLUMNS qualifier.

/BACKUP

Modifies the time value specified with the /BEFORE or /SINCE qualifier. /BACKUP selects files according to the dates of their most recent backups. This qualifier is incompatible with the other qualifiers that also allow you to select files according to time attributes: /CREATED, /EXPIRED, and /MODIFIED. If you specify none of these four time qualifiers, the default is /CREATED.

/BEFORE[=time]

Selects only those files dated prior to the specified time. You can specify time as an absolute time, as a combination of absolute and delta times, or as one of the following keywords: TODAY (default), TOMORROW, or YESTERDAY. Specify one of the following qualifiers with /BEFORE to indicate the time attribute to be used as the basis for selection: /BACKUP, /CREATED (default), /EXPIRED, or /MODIFIED.

Time must be specified using the VMS format of DD-MMM-YYY HH:MM:SS.CC; international date/time formats will not be accepted by the system. See Section 1.4 of the *VMS DCL Concepts Manual* for complete information on specifying time values.

/BRIEF (default)

Displays only a file's name, type, and version number. The brief format lists the files in alphabetical order from left to right on each line, in descending version number order. You can use the /ACL, /DATE, /FILE_ID, /NOHEADING, /OWNER, /PROTECTION, /SECURITY, and /SIZE qualifiers to expand a brief display.

/BY_OWNER[=uic]

Selects only those files whose owner user identification code (UIC) matches the specified owner UIC. The default UIC is that of the current process.

Specify the UIC using standard UIC format as described in Section 8.1 of the *VMS DCL Concepts Manual.*

/COLUMNS=n

Specifies the number of columns in a brief display. The default is four. However, you can request as many columns as you like, restricted by the value of the /WIDTH qualifier. The /COLUMNS qualifier is incompatible with /ACL, /FULL, and /SECURITY.

The number of columns actually displayed depends on the amount of information requested for each column and the DISPLAY value of the /WIDTH qualifier. The system displays only as many columns as can

fit within the default or specified display width, regardless of how many columns you specify with /COLUMNS.

The DIRECTORY command truncates long file names only when you have asked for additional information to be included in each column. The default file name size is 19. Use the /WIDTH qualifier to change the default. When a file name is truncated, the system displays one less character than the file name field size and inserts a vertical bar in the last position. For example, if the file name is SHOW_QUEUE_CHARACTERISTICS, and if you requested DIRECTORY to display both file name and size in each column, the display for that file would be SHOW_QUEUE_CHARACT| 120.

/CREATED (default)

Modifies the time value specified with the /BEFORE or /SINCE qualifier. /CREATED selects files based on their dates of creation. This qualifier is incompatible with the other qualifiers that also allow you to select files according to time attributes: /BACKUP, /EXPIRED, and /MODIFIED. If you specify none of these four time qualifiers, the default is /CREATED.

/DATE[=option]
/NODATE (default)

Includes the backup, creation, expiration, or modification date for each specified file; the default is /NODATE. If you use the /DATE qualifier without an option, the creation date is provided. Possible options are as follows:

ALL	Creation, expiration, backup, and last modification dates
BACKUP	Last backup date
CREATED	Creation date
EXPIRED	Expiration date
MODIFIED	Last modification date

/EXCLUDE=(file-spec[,...])

Excludes the specified files from the DIRECTORY operation. When using /EXCLUDE in a DIRECTORY operation of a different device, use only the file name in the file specification. Wildcard characters are allowed in the file specification. However, you cannot use relative version numbers to exclude a specific version. If you provide only one file specification, you can omit the parentheses.

/EXPIRED

Modifies the time value specified with the /BEFORE or /SINCE qualifier. /EXPIRED selects files according to their expiration dates. (The expiration date is set with the SET FILE/EXPIRATION_DATE command.) The /EXPIRED qualifier is incompatible with the other qualifiers that also allow you to select files according to time attributes: /BACKUP, /CREATED, and /MODIFIED. If you specify none of these four time qualifiers, the default is /CREATED.

/FILE_ID

Controls whether the file's identification number (FID) is displayed. By default, a file's identification is not displayed unless the /FULL qualifier is specified.

/FULL

Displays the following information for each file:

- File name
- File type
- Version number
- Number of blocks used
- Number of blocks allocated
- Date of creation
- Date last modified and revision number
- Date of expiration
- Date of last backup
- File owner's UIC
- File protection
- File identification number (FID)
- File organization
- Journaling information
- Other file attributes
- Record attributes
- Record format
- Access control list (ACL)

/GRAND_TOTAL

Displays only the totals for all files and directories that have been specified. Suppresses both the per-directory total and individual file information. (See the /TRAILING qualifier for information on displaying directory totals.)

/HEADING
/NOHEADING

Controls whether heading lines consisting of a device description and directory specification are printed. The default output format provides this heading. When /NOHEADING is specified, the display is in single-column format and the device and directory information appears with each file name. The /NOHEADING qualifier overrides /COLUMNS.

The combination of the /NOHEADING and /NOTRAILING qualifiers is useful in command procedures where you want to create a list of complete file specifications for later operations.

/MODIFIED

Modifies the time value specified with the /BEFORE or /SINCE qualifier. /MODIFIED selects files according to the dates on which they were last modified. This qualifier is incompatible with the other qualifiers that also allow you to select files according to time attributes: /BACKUP, /CREATED, and /EXPIRED. If you specify none of these four time modifiers, the default is /CREATED.

/OUTPUT[=file-spec]
/NOOUTPUT

Controls where the output of the command is sent. By default, the display is written to the current SYS$OUTPUT device. No wildcard characters are allowed.

If you enter /OUTPUT with a partial file specification (for example, /OUTPUT=[JONES]), DIRECTORY is the default file name and LIS the default file type. If you enter /NOOUTPUT, output is suppressed.

If the output will be written to a file in the same directory, the output file name will appear in the directory listing.

/OWNER
/NOOWNER (default)

Controls whether the file owner's UIC is listed.

The default size of the owner field is 20 characters. If the file owner's UIC exceeds the length of the owner field, the information will be truncated. The size of this field can be altered by specifying /WIDTH=OWNER, along with a value for the OWNER field. For more information, see the description of the /WIDTH qualifier.

/PRINTER

Puts the display in a file and queues the file to SYS$PRINT for printing under the name given by the /OUTPUT qualifier. If you do not specify the /OUTPUT qualifier, output is directed to a temporary file named DIRECTORY.LIS, which is queued for printing and then deleted.

/PROTECTION
/NOPROTECTION (default)

Controls whether the file protection for each file is listed.

/SECURITY

Controls whether information about file security is displayed; using /SECURITY is equivalent to using the /ACL, /OWNER, and /PROTECTION qualifiers together.

/SELECT=(keyword[,...])

Allows you to select files for display according to size. Choose one of the following keywords:

SIZE=MAXIMUM=n	Displays files that have fewer blocks than the value of n, which defaults to 1,073,741,823. Use with MINIMUM=n to specify a size range for files to be displayed.
SIZE=MINIMUM=n	Displays files that have blocks equal to or greater than the value of n, which defaults to 0. Use with MAXIMUM=n to specify a size range for files to be displayed.
SIZE=(MAXIMUM=n,MINIMUM=m)	Displays files whose blocksize falls within the specified MAXIMUM and MINIMUM range.

By default, file selection is based on other criteria.

/SINCE[=time]

Selects only those files dated after the specified time. You can specify time as an absolute time, a combination of absolute and delta times, or as one of the following keywords: TODAY (default), TOMORROW, or YESTERDAY. Specify one of the following qualifiers with /BEFORE to indicate the time attribute to be used as the basis for selection: /BACKUP, /CREATED (default), /EXPIRED, or /MODIFIED.

Time must be specified using the VMS format of DD-MMM-YYY HH:MM:SS.CC; international date/time formats will not be accepted by the system. See Section 1.4 of the *VMS DCL Concepts Manual* for complete information on specifying time values.

/SIZE[=option]
/NOSIZE (default)

Displays the size in blocks of each file. If you omit the option parameter, the default lists the file size in blocks used (USED). Specify one of the following options:

ALL	Lists the file size both in blocks allocated and blocks used
ALLOCATION	Lists the file size in blocks allocated
USED	Lists the file size in blocks used

The size of this field can be altered by supplying the SIZE value of the /WIDTH qualifier.

/TOTAL

Displays only the directory name and total number of files.

By default, the output format is /BRIEF, which gives this total, but also lists all the file names, file types, and their version numbers.

/TRAILING
/NOTRAILING

Controls whether trailing lines that provide the following summary information are displayed:

- Number of files listed
- Total number of blocks used per directory
- Total number of blocks allocated
- Total number of directories and total blocks used or allocated in all directories (only if more than one directory is listed)

By default, the output format includes most of this summary information. The /SIZE and /FULL qualifiers determine more precisely what summary information is included.

Used by itself, /TRAILING lists the number of files in the directory. Used with /SIZE, /TRAILING lists the number of files and the number of blocks (displayed according to the option of the /SIZE qualifier, FULL or ALLOCATION). Used with /FULL, /TRAILING lists the number of files as well as the number of blocks used and allocated. If more than one directory is listed, the summary includes the total number of directories, the total number of blocks used, and the total number of blocks allocated.

/VERSIONS=n

Specifies the number of versions of a file to be listed. The default is all versions of each file. A value less than 1 is not allowed.

/WIDTH=(keyword[,...])

Formats the width of the display. If you specify only one keyword, you can omit the parentheses. Possible keywords are as follows:

DISPLAY=n	Specifies the total width of the display as an integer in the range 1 through 256 and defaults to 0 (setting the display width to the terminal width). If the total width of the display exceeds the terminal width, the information will be truncated.
FILENAME=n	Specifies the width of the file name field; defaults to 19. If you have requested another piece of information to be displayed along with the file name in each column, file names that exceed the n parameter cause the line to wrap, after the file name field. (See the /COLUMNS qualifier.)
OWNER=n	Specifies the width of the owner field; defaults to 20. If the owner's UIC exceeds the length of the owner field, the information will be truncated.
SIZE=n	Specifies the width of the size field; defaults to 6. If the file size exceeds the length of the size field, the information will be truncated.

EXAMPLES

1

```
$ DIRECTORY AVERAGE.*

Directory DISK$DOCUMENT:[MALCOLM]

AVERAGE.EXE;6      AVERAGE.FOR;6      AVERAGE.LIS;4      AVERAGE.OBJ;12

Total of 4 files.
```

In this example, the DIRECTORY command lists all files with the file name AVERAGE and any file type.

2

```
$ DIRECTORY/SIZE=USED/DATE=CREATED/VERSIONS=1/PROTECTION  AVERAGE

Directory DISK$DOCUMENT:[MALCOLM]

AVERAGE.EXE;6       6        10-APR-1988 15:43 (RWED,RWED,RWED,RE)
AVERAGE.FOR;6       2         2-APR-1988 10:29 (RWED,RWED,RWED,RE)
AVERAGE.LIS;4       5         9-APR-1988 16:27 (RWED,RWED,RWED,RE)
AVERAGE.OBJ;6       2         9-APR-1988 16:27 (RWED,RWED,RWED,RE)

Total of 4 files, 15 blocks.
```

In this example, the DIRECTORY command lists the number of blocks used, the creation date, and the file protection code for the highest version number of all files named AVERAGE in the current default directory.

3

```
$ DIRECTORY/FULL [JONES.ITALIA]PROJECTIONS.LIS

Directory WORK:[JONES.ITALIA]

PROJECTIONS.LIS;1           File ID:  (7449,36222,2)
Size:           21/21         Owner:    [DOC,JONES]
Created:   5-MAY-1988 15:49:03.11
Revised:   5-MAY-1988 15:49:49.39 (2)
Expires:   <None specified>
Backup:    <No backup recorded>
File organization:  Sequential
File attributes:    Allocation: 21, Extend: 0, Global buffer count: 0,
                    No version limit
Record format:      Variable length, maximum 80 bytes
Record attributes:  Carriage return carriage control
Journaling enabled: None
File protection:    System:RWED, Owner:RWED, Group:RE, World:
Access Cntrl List:  None

Total of 1 file, 21/21 blocks.
```

The DIRECTORY command in this example shows the date/time format using the default language, English, and the default VMS format. Users also may select other languages and formats that have been defined on their systems with Run-Time Library international date/time formatting routines. See the *VMS RTL Library (LIB$) Manual.*

4

```
$ DIRECTORY/VERSIONS=1/COLUMNS=1 AVERAGE.*
```

The DIRECTORY command in this example lists only the highest version of each file named AVERAGE in the current default directory. The format is brief and restricted to one column. Heading and trailing lines are provided.

5

```
$ DIRECTORY BLOCK%%%
```

The DIRECTORY command in this example locates all versions and types of files in the default device and directory whose names begin with the letters BLOCK and end with any three additional characters. The default output format is brief, four columns, with heading and trailing lines.

6

```
$ DIRECTORY/EXCLUDE=(AVER.DAT;*,AVER.EXE;*) [*...]AVER
```

The DIRECTORY command in this example lists and totals all versions and types of files named AVER in all directories and subdirectories on the default disk, except any files named AVER.DAT and AVER.EXE.

7

```
$ DIRECTORY/SIZE=ALL FRESNO::DISK1:[TAYLOR]*.COM
```

This DIRECTORY command in this example lists all versions of all files with the file type COM in the directory TAYLOR on node FRESNO and device DISK1. The listing includes the file size both in blocks used and in blocks allocated for each file.

DIRECTORY

8
```
$ DIRECTORY-
_$/MODIFIED/SINCE=31-DEC-1988:01:30/SIZE=ALL/OWNER-
_$/PROTECTION/OUTPUT=UPDATE/PRINTER [A*]
```

The DIRECTORY command in this example locates all files that have been modified since 1:30 A.M. on December 31, 1988 and that reside on the default disk in all directories whose names begin with the letter A. It formats the output to include all versions, the size used and size allocated, the date last modified, the owner, and the protection codes. The output is directed to a file named UPDATE.LIS that is queued automatically to the default printer queue and then deleted when done.

DISCONNECT

Breaks the connection between a physical terminal and a virtual terminal. After the physical terminal is disconnected, both the virtual terminal and the process using it remain on the system.

Requires that your physical terminal is connected to a virtual terminal.

FORMAT

DISCONNECT

PARAMETERS

None.

DESCRIPTION

Use the DISCONNECT command to disconnect a physical terminal from a virtual terminal and its associated process. The virtual terminal and the process remain on the system, so you can use the CONNECT command to reconnect to the process later. (See the description of the CONNECT command for more information about virtual terminals and how to connect to them.) To terminate a process connected to a virtual terminal, use the LOGOUT command.

After you are disconnected from a virtual terminal, you can use the physical terminal to log in again.

You can use the DISCONNECT command only if your physical terminal is connected to a virtual terminal.

QUALIFIER

/CONTINUE
/NOCONTINUE (default)

Controls whether the CONTINUE command is executed in the current process just before connecting to another process. This permits an interrupted image to continue processing after the disconnect takes place.

EXAMPLES

1

```
$ DISCONNECT
```

This command disconnects a physical terminal from a virtual terminal, but does not log the process out. The physical terminal can now be used to log in again.

```
2  $ RUN PAYROLL
   ^Y
   $ DISCONNECT/CONTINUE
```

In this example, the RUN command is issued from a physical terminal that is connected to a virtual terminal. After the image PAYROLL.EXE is interrupted, the DISCONNECT command disconnects the physical and the virtual terminals without logging out the process. The /CONTINUE qualifier allows the image PAYROLL.EXE to continue to execute. However, the terminal can be used to log in again and perform other work.

DISMOUNT

Closes a mounted disk or magnetic tape volume for further processing and deassigns the logical name associated with the device. If the volume is mounted with the /SHARE qualifier, its logical name is deassigned but the volume remains mounted until all processes using it dismount it or terminate. Note that all open files on the volume must be closed before the actual dismount can be done. Note, also, that the file system cannot dismount a volume while any known file lists associated with it contain entries.

Requires the GRPNAM and SYSNAM user privileges to dismount group and system volumes.

FORMAT

DISMOUNT *device-name[:]*

PARAMETER

device-name[:]

Name of the device containing the volume — either a logical name or a physical name. If a physical name is specified, the controller defaults to A and the unit defaults to 0.

If the volume currently mounted on the device is a member of a disk or tape volume set, all volumes in the set are dismounted, unless the /UNIT qualifier is specified.

DESCRIPTION

The DISMOUNT command (which invokes the $DISMOU system service) performs the following operations:

- Removes the volume from the user's list of mounted volumes, deletes the logical name (if any) associated with the volume, and decrements the mount count.
- If the mount count equals 0 after being decremented, DISMOUNT marks the volume for dismounting, and then waits until the volume is idle before dismounting it. (A native volume is idle when no user has an open file to the volume; a foreign volume is idle when no channels are assigned to the volume.)
- If the mount count does not equal 0 after being decremented, DISMOUNT does not mark the volume for dismount (because the volume must have been mounted shared). In this case, the total effect for the issuing process is that the process is denied access to the volume and a logical name entry is deleted.
- After a volume is actually dismounted, nonpaged pool is returned to the system. Paged pool is also returned if the volume had been mounted using the /GROUP or /SYSTEM qualifiers.
- If a volume is part of a Files–11 volume set and the flag bit DMT$V_UNIT is not set, the entire volume set will be dismounted.

If the volume was mounted with the /SHARE qualifier, it is not actually dismounted until all users who mounted it dismount it or log off. However, the DISMOUNT command deassigns the logical name associated with the device.

If the device was allocated with an ALLOCATE command, it remains allocated after the volume is dismounted with the DISMOUNT command. If the device was implicitly allocated by the MOUNT command, the DISMOUNT command deallocates it.

If the volume was mounted with the /GROUP or the /SYSTEM qualifier, it is dismounted even if other users are currently accessing it. The GRPNAM and SYSNAM user privileges are required to dismount group and system volumes, respectively.

Note that the file system performs volume dismounting, and all open files on the volume must be closed for the dismount to finish. Note also, that the file system cannot dismount a volume while any known file lists associated with it contain entries. Thus, a substantial amount of time can pass between the time you enter the DISMOUNT command and the completion of the dismount. Always wait for the drive to unload before you remove the volume. (To verify that the dismount has completed, enter the SHOW DEVICES command.)

QUALIFIERS

/ABORT

Requires volume ownership or the user privilege VOLPRO to use this qualifier with a volume that is mounted neither group nor system.

Specifies that the volume is to be dismounted, regardless of who actually mounted it. The primary purpose of the /ABORT qualifier is to terminate mount verification. DISMOUNT/ABORT also cancels any outstanding I/O requests. If the volume was mounted with the /SHARE qualifier, the /ABORT qualifier causes the volume to be dismounted for all of the users who mounted it.

/CLUSTER

Dismounts a volume clusterwide. After the DISMOUNT command successfully dismounts the volume on the local node, the volume is dismounted on every other node in the existing VAXcluster. If the system is not a member of a VAXcluster, the /CLUSTER qualifier has no effect.

/UNIT

Dismounts only the volume of a volume set on the specified device. By default, all volumes in a set are dismounted.

Note: Avoid dismounting the root volume of a volume set, because it contains the master file directory (MFD). It may be impossible to access files on a volume set if the MFD is not accessible.

/UNLOAD (default)
/NOUNLOAD

Unloads the device on which the volume is mounted. If you specify /NOUNLOAD, the device remains in a ready state.

EXAMPLES

1
```
$ MOUNT MT: PAYVOL TAPE
    .
    .
    .
$ DISMOUNT TAPE:
```

The MOUNT command in this example mounts the tape whose volume identification is PAYVOL on the device MTA0: and assigns the logical name TAPE to the device. By default, the volume is not shareable. The DISMOUNT command releases access to the volume, deallocates the device, and deletes the logical name TAPE.

2
```
$ MOUNT/SHARE DBA3:  DOC_FILES
    .
    .
    .
$ DISMOUNT DBA3:
```

The MOUNT command in this example mounts the volume labeled DOC_FILES on the device DBA3. Other users can enter MOUNT commands to access the device. The DISMOUNT command shown in this example deaccesses the device for the process issuing the command. If other users still have access to the volume, the volume remains mounted for their process or processes.

3
```
$ DISMOUNT/NOUNLOAD  DMA2:
```

The DISMOUNT command in this example dismounts the volume; the /NOUNLOAD qualifier requests that the volume remain in a ready state.

4
```
$ MOUNT/BIND=PAYROLL  DMA1:,DMA2:  PAYROLL01,PAYROLL02
    .
    .
    .
$ DISMOUNT/UNIT  DMA2:
```

The MOUNT command in this example mounts PAYROLL, a two-volume set. The DISMOUNT command dismounts only PAYROLL02, leaving PAYROLL01 accessible. Note that because the master file directory (MFD) for the volume set is on the root volume, you should not dismount the root volume (in this case, PAYROLL01) of the volume set.

DUMP

Displays the contents of a file, disk volume, or magnetic tape volume in decimal, hexadecimal, or octal format, as well as the ASCII conversion.

FORMAT

DUMP *file-spec [,...]*

PARAMETER

file-spec

Specifies the file or name of the device being dumped.

If the specified device is not a disk, tape, or network device, or if the device is mounted with the /FOREIGN qualifier, the file specification must contain only the device name.

If the specified device is a network device, a disk device, or tape device that is mounted without the /FOREIGN qualifier, the file specification can contain wildcards.

DESCRIPTION

By default, the DUMP command formats the output both in ASCII characters and in hexadecimal longwords. You can specify another format for the dump by using a radix qualifier (/OCTAL, /DECIMAL, or /HEXADECIMAL) or a length qualifier (/BYTE, /WORD, or /LONGWORD).

Dumping Files

If the input medium is a network device, a disk device, or tape device that is mounted without the /FOREIGN qualifier, the DUMP command operates on files. You can dump files by either records or blocks. Wildcard specifications can be used to select a group of files for processing.

Dumping Volumes

If the input medium is not a disk or tape device, or if it is mounted with the /FOREIGN qualifier, the DUMP command operates on the input device as a non-file-structured medium. Disk devices are dumped by 512-byte logical blocks. Other devices are dumped by physical blocks. No repositioning of the input medium occurs; therefore, consecutive blocks on a tape can be dumped by a single DUMP command.

If you have LOG_IO (logical I/O) privilege, you can dump random blocks on a Files-11 volume. For example, using the /BLOCKS qualifier you could dump block 100 on the system disk.

Reading Dumps

The ASCII representation is read left to right. The hexadecimal, decimal, and octal representations are read right to left.

Specifying Numeric Qualifier Values

The numeric values for the /BLOCKS, /RECORDS, and /NUMBER qualifiers can be specified either as decimal numbers or with a leading %X, %O, or %D to signify hexadecimal, octal, or decimal numbers respectively. For example, the following are all valid ways to specify decimal value 24:

```
24
%X18
%O30
%D24
```

QUALIFIERS

/ALLOCATED

Includes in the dump all blocks allocated to the file. (By default, the dump does not include blocks following the end-of-file.)

You can specify /ALLOCATED if the input is a disk that is mounted without the /FOREIGN qualifier. /ALLOCATED and /RECORDS are mutually exclusive.

/BLOCKS[=(option[,...])]

Dumps the specified blocks one block at a time, which is the default method for all devices except network devices.

Block numbers are specified as integers relative to the beginning of the file. Typically, blocks are numbered beginning with 1. If a disk device is mounted /FOREIGN, blocks are numbered beginning with 0. Select a range of blocks to be dumped by specifying one of the following options:

START:n	Specifies the number of the first block to be dumped; the default is the first block.
END:n	Specifies the number of the last block to be dumped; the default is the last block or the end-of-file block, depending on the /ALLOCATED qualifier.
COUNT:n	Specifies the number of files to be dumped. COUNT provides an alternative to END; you may not specify both.

If you specify only one option, you can omit the parentheses.

/BLOCKS and /RECORDS are mutually exclusive.

Use the /BLOCKS qualifier to dump random blocks from Files-11 volumes. This requires LOG-IO (logical I/O) privilege.

/BYTE

Formats the dump in bytes. /BYTE, /LONGWORD, and /WORD are mutually exclusive. The default format is composed of longwords.

/DECIMAL

Dumps the file in decimal radix. /DECIMAL, /HEXADECIMAL (default), and /OCTAL are mutually exclusive.

/FILE_HEADER

Dumps each data block that is a valid Files-11 header in Files-11 header format rather than the selected radix and length.

/FORMATTED (default)
/NOFORMATTED

Dumps the file header in Files-11 format; /NOFORMATTED dumps the file header in octal format. This qualifier is useful only when /HEADER is specified.

/HEADER

Dumps the file header and access control list. To dump only the file header, and not the file contents, also specify /BLOCK=(COUNT:0). /HEADER is invalid for devices mounted /FOREIGN.

Use the /FORMATTED qualifier to control the format of the display.

You can use the /FILE_HEADER qualifier with /HEADER to have Files-11 file headers printed in an interpreted representation.

By default, the file header is not displayed.

/HEXADECIMAL (default)

Dumps the file in hexadecimal radix. /DECIMAL, /HEXADECIMAL (default), and /OCTAL are mutually exclusive.

/LONGWORD (default)

Formats the dump in longwords. /BYTE, /LONGWORD, and /WORD are mutually exclusive.

/NUMBER[=n]

Specifies how byte offsets are assigned to the lines of output. If you specify /NUMBER, the byte offsets increase continuously through the dump, beginning with n; if you omit /NUMBER, the first byte offset is 0. By default, the byte offset is reset to 0 at the beginning of each block or record.

/OCTAL

Dumps the file in octal radix. /DECIMAL, /HEXADECIMAL (default), and /OCTAL are mutually exclusive.

/OUTPUT[=file-spec]

Specifies the output file for the dump. If you do not specify a file specification, the default is the file name of the file being dumped and the file type DMP. If /OUTPUT is not specified, the dump goes to SYS$OUTPUT. No wildcard characters are allowed. /OUTPUT and /PRINTER are mutually exclusive.

/PRINTER

Queues the dump to SYS$PRINT in a file named with the file name of the file being dumped and the file type DMP. If /PRINTER is not specified, the dump goes to SYS$OUTPUT. No wildcard characters are allowed. /OUTPUT and /PRINTER are mutually exclusive.

/RECORDS[=(option[,...])]

Dumps the file a record at a time rather than a block at a time. (By default, input is dumped one block at a time for all devices except network devices.)

Blocks are numbered beginning with 1.

Select a range of blocks to be dumped by specifying one of the following options:

START:n	Specifies the number of the first record to be dumped; the default is the first record.
END:n	Specifies the number of the last record to be dumped; the default is the last record of the file.
COUNT:n	Specifies the number of records to be dumped. COUNT provides an alternative to END; you may not specify both.

If you specify only one option, you can omit the parentheses.

If you specify /RECORDS, you cannot specify /ALLOCATED or /BLOCKS.

/WORD

Formats the dump in words. /BYTE, /LONGWORD, and /WORD are mutually exclusive.

EXAMPLES

1

```
$ DUMP TEST.DAT
Dump of file DISK0:[NORMAN]TEST.DAT;1 on 15-APR-1988 15:43:26.08
File ID (3134,818,2)   End of file block 1 / Allocated 3
Virtual block number 1 (00000001), 512 (0200) bytes
 706D6173 20612073 69207369 68540033 3.This is a samp 000000
 73752065 62206F74 20656C69 6620656C le file to be us 000010
 61786520 504D5544 2061206E 69206465 ed in a DUMP exa 000020
 00000000 00000000 0000002E 656C706D mple..........  000030
 00000000 00000000 00000000 00000000 ................ 000040
 00000000 00000000 00000000 00000000 ................ 000050
 00000000 00000000 00000000 00000000 ................ 000060
          .
          .
          .
 00000000 00000000 00000000 00000000 ................ 0001E0
 00000000 00000000 00000000 00000000 ................ 0001F0
```

The DUMP command displays the contents of TEST.DAT both in hexadecimal longword format and in ASCII beginning with the first block in the file.

2
```
$ DUMP TEST.DAT/OCTAL/BYTE
Dump of file DISKO:[NORMAN]TEST.DAT;1 on 15-APR-1988 15:45:33.58
File ID (74931,2,1)   End of file block 1 / Allocated 3
Virtual block number 1 (00000001), 512 (0200) bytes
 151 040 163 151 150 124 000 063 3.This i 000000
 160 155 141 163 040 141 040 163 s a samp 000010
 040 145 154 151 146 040 145 154 le file  000020
 163 165 040 145 142 040 157 164 to be us 000030
 040 141 040 156 151 040 144 145 ed in a  000040
 141 170 145 040 120 115 125 104 DUMP exa 000050
 377 377 000 056 145 154 160 155 mple.... 000060
 000 000 000 000 000 000 000 000 ........ 000070
 000 000 000 000 000 000 000 000 ........ 000100
 000 000 000 000 000 000 000 000 ........ 000110
          .
          .
          .
 000 000 000 000 000 000 000 000 ........ 000760
 000 000 000 000 000 000 000 000 ........ 000770
```

The DUMP command displays the image of the file TEST.DAT, formatted both in octal bytes and in ASCII characters beginning with the first block.

3
```
$ DUMP NODE3::DISK2:[STATISTICS]RUN1.DAT
```

This command line dumps the file RUN1.DAT that is located at remote node NODE3. The default DUMP format will be used.

EDIT/ACL

Invokes the Access Control List (ACL) Editor to create or modify an access control list for a specified object. For a complete description of the VMS Access Control List (ACL) Editor, including information about the EDIT /ACL command and its qualifiers, see the *VMS Access Control List Editor Manual*. The /ACL qualifier is required.

FORMAT

EDIT/ACL *object-spec*

EDIT/EDT

Invokes the VAX EDT interactive text editor. The /EDT qualifier is not required, because EDT is the VMS default editor.

FORMAT

EDIT *file-spec*

PARAMETER

file-spec

Specifies the file to be created or edited using the EDT editor. If the file does not exist, it is created by EDT.

The EDT editor does not provide a default file type when creating files; if you do not include a file type, it is null. The file must be a disk file on a Files–11 formatted volume.

No wildcard characters are allowed in the file specification.

DESCRIPTION

The EDT editor creates or edits text files. You can use EDT to enter or edit text in three modes: keypad, line, or nokeypad. Keypad editing, which is screen-oriented, is available on VT300-series, VT200-series, VT100, and VT52 terminals. A screen-oriented editor allows you to see several lines of text at once and move the cursor throughout the text in any direction. Line editing operates on all terminals. In fact, if you have a terminal other than a VT300-series, VT200-series, VT100, or VT52, line editing is the only way you can use EDT. You might prefer line editing if you are accustomed to editing by numbered lines. Nokeypad mode is a command oriented screen editor available on VT300-series, VT200-series, VT100, and VT52 terminals. You can use line mode and nokeypad mode to redefine keys for use in keypad mode.

When you invoke EDT, you are in line mode by default. If you are editing an existing file, EDT displays the line number and text for the first line of the file. If you are creating a new file, EDT displays the following message:

```
Input file does not exist
[EOB]
```

In either case, EDT then displays the line mode prompt, which is the asterisk (*).

For complete details on the EDT editor, see the *VAX EDT Reference Manual.*

QUALIFIERS

/COMMAND[=file-spec]
/NOCOMMAND

Determines whether or not EDT uses a startup command file. The /COMMAND file qualifier should be followed by an equal sign and the specification of the command file. The default file type for command files is EDT. No wildcard characters are allowed in the file specification.

The following command line invokes EDT to edit a file named MEMO.DAT and specifies that EDT use a startup command file named XEDTINI.EDT:

```
$ EDIT/COMMAND=XEDTINI.EDT MEMO.DAT
```

If you do not include the /COMMAND=command file qualifier, EDT looks for the EDTSYS logical name assignment. If EDTSYS is not defined, EDT processes the systemwide startup command file SYS$LIBRARY:EDTSYS.EDT. If this file does not exist, EDT looks for the EDTINI logical name assignment. If EDTINI is not defined, EDT looks for the file named EDTINI.EDT in your default directory. If none of these files exists, EDT begins your editing session in the default state.

To prevent EDT from processing either the systemwide startup command file or the EDTINI.EDT file in your default directory, use the /NOCOMMAND qualifier as follows:

```
$ EDIT/NOCOMMAND MEMO.DAT
```

/CREATE (default)
/NOCREATE

Controls whether EDT creates a new file when the specified input file is not found.

Normally, EDT creates a new file to match the input file specification if it cannot find the requested file name in the specified directory. When you use /NOCREATE in the EDT command line and type a specification for a file that does not exist, EDT displays an error message and returns to the DCL command level as follows:

```
$ EDIT/NOCREATE NEWFILE.DAT
Input file does not exist
$
```

/JOURNAL[=journal-file]
/NOJOURNAL

Determines whether EDT keeps a journal file during your editing session. A journal file contains a record of the keystrokes you enter during an editing session. The default file name for the journal file is the same as the input file name. The default file type is JOU. The /JOURNAL qualifier enables you to use a different file specification for the journal file.

The following command line invokes EDT to edit a file named MEMO.DAT and specifies the name SAVE.JOU for the journal file:

```
$ EDIT/JOURNAL=SAVE MEMO.DAT
```

If you are editing a file from another directory and want the journal file to be located in that directory, you must use the /JOURNAL qualifier with a file specification that includes the directory name. Otherwise, EDT creates the journal file in the default directory.

The directory that is to contain the journal file should not be write-protected.

To prevent EDT from keeping a record of your editing session, use the /NOJOURNAL qualifier in the EDT command line as follows:

```
$ EDIT/NOJOURNAL MEMO.DAT
```

Once you have created a journal file, use EDT/RECOVER to execute the commands in the journal file. No wildcard characters are allowed in the file specification.

/OUTPUT=output-file
/NOOUTPUT

Determines whether EDT creates an output file at the end of your editing session. The default file specification for both the input file and the output file is the same. Use the /OUTPUT qualifier to give the output file a different file specification from the input file.

The following command line invokes EDT to edit a file named MEMO.DAT and gives the resulting output file the name OUTMEM.DAT:

```
$ EDIT/OUTPUT=OUTMEM.DAT MEMO.DAT
```

You can include directory information as part of your output file specification to send output to another directory as follows:

```
$ EDIT/OUTPUT=[BARRETT.MAIL]MEMO.DAT MEMO.DAT
```

The /NOOUTPUT qualifier suppresses the creation of an output file, but not the creation of a journal file. If you decide that you do not want an output file, you can use /NOOUTPUT as follows:

```
$ EDIT/NOOUTPUT MEMO.DAT
```

A system interruption does not prevent you from recreating your editing session because a journal file is still being maintained. To save your editing session, even when you specify /NOOUTPUT, use the line mode command WRITE to put the text in an external file before you end the session.

No wildcard characters are allowed in the file specification.

/READ_ONLY
/NOREAD_ONLY (default)

Determines whether EDT keeps a journal file and creates an output file. With the default /NOREAD_ONLY, EDT maintains the journal file and creates an output file when it processes the line mode command EXIT. Using the /READ_ONLY qualifier has the same effect as specifying both the /NOJOURNAL and /NOOUTPUT qualifiers.

The following command line invokes EDT to edit a file named CALENDAR.DAT, but does not create a journal file or an output file:

```
$ EDIT/READ_ONLY CALENDAR.DAT
```

Use /READ_ONLY when you are searching a file and do not intend to make any changes to it. To modify the file, use the line mode command WRITE to save your changes. Remember, however, that you have no journal file.

/RECOVER
/NORECOVER (default)

Determines whether or not EDT reads a journal file at the start of the editing session.

When you use the /RECOVER qualifier, EDT reads the appropriate journal file and processes whatever commands it contains. The appropriate syntax is as follows:

```
$ EDIT/RECOVER MEMO.DAT
```

If the journal file type is not JOU or the file name is not the same as the input file name, you must include both the /JOURNAL qualifier and the /RECOVER qualifier as follows:

```
$ EDIT/RECOVER/JOURNAL=SAVE.XXX MEMO.DAT
```

Because /NORECOVER is the default for EDT, you do not need to specify it in a command line.

EXAMPLES

1

```
$ EDIT/OUTPUT=NEWFILE.TXT OLDFILE.TXT
    1       This is the first line of the file OLDFILE.TXT.
*
```

This EDIT command invokes the EDT editor to edit the file OLDFILE.TXT. EDT looks for the EDTSYS logical name assignment. If EDTSYS is not defined, EDT processes the systemwide startup command file SYS$LIBRARY:EDTSYS.EDT. If this file does not exist, EDT looks for the EDTINI logical name assignment. If EDTINI is not defined, EDT looks for the file named EDTINI.EDT in your default directory. If none of these files exists, EDT begins your editing session in the default state. When the session ends, the edited file has the name NEWFILE.TXT.

2

```
$ EDIT/RECOVER OLDFILE.TXT
```

This EDIT command invokes the EDT editor to recover from an abnormal exit during a previous editing session. EDT opens the file OLDFILE.TXT, and then processes the journal file OLDFILE.JOU. Once the journal file has been processed, the user can resume interactive editing.

EDIT/FDL

Invokes the VMS FDL Editor (EDIT/FDL) to create and modify File Definition Language (FDL) files. The /FDL qualifier is required. For a complete description of the FDL Utility, including more information about the EDIT/FDL command and its qualifiers, see the FDL Facility in the *VMS File Definition Language Facility Manual*.

FORMAT

EDIT/FDL *file-spec*

EDIT/SUM

Invokes the SUMSLP batch-oriented editor, to update a single input file with multiple files of edit commands.

For a complete description of the SUMSLP editor, including information about the EDIT/SUM command and its qualifiers, see the *VMS SUMSLP Utility Manual*.

FORMAT

EDIT/SUM *input-file*

EDIT/TECO

Invokes the TECO interactive text editor. The /TECO qualifier is required.

FORMAT

EDIT/TECO *[file-spec]*
EDIT/TECO/EXECUTE=command-file *[argument]*

PARAMETER

file-spec

Specifies the file to be created or edited using the TECO editor. If the file does not exist, it is created by TECO, unless you specify /NOCREATE. No wildcard characters are allowed in the file specification.

If you specify /MEMORY (default) without a file specification, TECO edits the file identified by the logical name TEC$MEMORY. If TEC$MEMORY has no equivalence string, or if /NOMEMORY is specified, TECO starts in command mode and does not edit an existing file.

If you specify /MEMORY and a file specification, the file specification is equated to the logical name TEC$MEMORY.

DESCRIPTION

The TECO editor creates or edits text files. For detailed information on the use of TECO, see the *PDP-11 TECO Editor Reference Manual.*

QUALIFIERS

/COMMAND[=file-spec]
/NOCOMMAND

Controls whether a startup command file is used. The /COMMAND file qualifier may be followed by an equal sign and the specification of the command file. The default file type for command files is TEC.

The following command line invokes TECO to edit a file named MEMO.DAT and specifies that TECO use a startup command file named XTECOINI.TEC:

```
$ EDIT/TECO/COMMAND=XTECOINI.TEC MEMO.DAT
```

If you do not include the /COMMAND qualifier, or if you enter /COMMAND without specifying a command file, TECO looks for the TEC$INIT logical name assignment. If TEC$INIT is not defined, no startup commands are executed.

The logical name TEC$INIT can equate either to a string of TECO commands or to a dollar sign followed by a file specification. If TEC$INIT translates to a string of TECO commands, the string is executed; if it translates to a dollar sign followed by a file specification, the contents of the file are executed as a TECO command string. For further information, see the *PDP-11 TECO Editor Reference Manual.*

To prevent TECO from using any startup command file, use the /NOCOMMAND qualifier as follows:

```
$ EDIT/TECO/NOCOMMAND MEMO.DAT
```

No wildcards are allowed in the file specification.

/CREATE (default)
/NOCREATE

Creates a new file when the specified input file cannot be found. If /MEMORY is specified and no input file is specified, the file created is the one specified by the logical name TEC$MEMORY. Normally, TECO creates a new file to match the input file specification if it cannot find the requested file name in the specified directory. When you use /NOCREATE in the TECO command line and type a specification for a file that does not exist, TECO displays an error message and returns you to the DCL command level. /EXECUTE is not compatible with /CREATE and /NOCREATE.

/EXECUTE=command-file [argument]

Invokes TECO and executes the TECO macro found in the command file. The argument, if specified, appears in the text buffer when macro execution starts. Blanks or special characters must be enclosed in quotes. For detailed information on the use of TECO macros, see the *PDP-11 TECO Editor Reference Manual.*

/EXECUTE is incompatible with /CREATE and /MEMORY.

/MEMORY (default)
/NOMEMORY

Specifies that the last file you edited with TECO, identified by the logical name TEC$MEMORY, will be the file edited if you omit the file specification to the EDIT/TECO command.

/OUTPUT=output-file
/NOOUTPUT (default)

Controls how the output file is named at the end of your editing session. By default, the output file has the same name as the input file but is given the next higher available version number. Use the /OUTPUT qualifier to give the output file a file specification different from the input file.

The following command line invokes TECO to edit a file named MEMO.DAT and gives the resulting output file the name OUTMEM.DAT:

```
$ EDIT/TECO/OUTPUT=OUTMEM.DAT MEMO.DAT
```

You can include directory information as part of your output file specification to send output to another directory as follows:

```
$ EDIT/TECO/OUTPUT=[BARRETT.MAIL]MEMO.DAT MEMO.DAT
```

No wildcard characters are allowed in the file specification.

/READ_ONLY
/NOREAD_ONLY (default)

Controls whether or not an output file is created. By default, an output file is created; /READ ONLY suppresses the creation of the output file.

EDIT/TECO

EXAMPLES

1 `$ EDIT/TECO/OUTPUT=NEWFILE.TXT OLDFILE.TXT`

This EDIT command invokes the TECO editor to edit the file OLDFILE.TXT. TECO looks for the TEC$INIT logical name assignment. If TEC$INIT is not defined, TECO begins the editing session without using a command file. When the session ends, the edited file has the name NEWFILE.TXT.

2 `$ EDIT/TECO/EXECUTE=FIND_DUPS "TEMP, ARGS, BLANK"`

In this example, the /EXECUTE qualifier causes the TECO macro contained in the file FIND_DUPS.TEC to be executed, with the argument string "TEMP, ARGS, BLANK" located in the text buffer.

EDIT/TPU

Invokes the VAX Text Processing Utility (VAXTPU), running the interactive text editor Extensible VAX Editor (EVE).

To use EVE with an EDT-style keypad, invoke EVE and then use the EVE command SET KEYPAD EDT. (This replaces the VAXTPU EDT Keypad Emulator.) Similarly, you use EVE with a WPS-style keypad.

FORMAT

EDIT/TPU *[file-spec]*

PARAMETER

file-spec

Optionally, specifies the input file to be created or edited.

- If you specify an input file on the command line, EVE creates an edit buffer with the same name as that file.
- If you do not specify an input file, EVE creates an empty buffer named MAIN. You can then use the EVE command GET FILE to specify the file you want to edit or create; also you can edit the buffer MAIN and, at the completion of your editing session, specify the output file by using the EVE command WRITE FILE.

VAXTPU does not have a default file type for creating output files. If you do not specify a file type, the file type is null. The file must be a disk file on a Files–11 formatted volume.

Processing wildcards in the file specification depends on the VAXTPU application you are using. With EVE, you can use wildcards in the input file specification. For example:

```
$ EDIT/TPU *.TXT
```

If more than one file matches your request, EVE displays a list of files with the same file type and prompts for more information.

DESCRIPTION

The VAX Text Processing Utility (VAXTPU) consists of a procedural language, a compiler, an interpreter, and an editor called the Extensible VAX Editor (EVE). Using EVE, you can do the following:

- Create and edit more than one file in an editing session
- Use multiple buffers and windows
- Define keys, including EDT or WPS keypad and LEARN sequences
- Execute DCL commands from within the editor
- Run DECspell for a selected range or an entire buffer
- Spawn subprocesses or attach to other processes
- Customize or extend the editor using VAXTPU procedures
- Save key definitions and extensions, by creating section files

- Use initialization files to set private defaults
- Get online HELP on EVE commands and VAXTPU statements
- Recover your edits in case of a system failure

For information on VAXTPU and EVE commands, see the *VAX Text Processing Utility Manual*. To learn how to use EVE, see the *Guide to VMS Text Processing*.

QUALIFIERS

/COMMAND[=command-file] (default)
/NOCOMMAND

Determines whether a user-written command file is used. User-written command files contain VAXTPU procedures and statements to extend or modify the editor or to define a special text-processing environment for creating your own application. The default file type is TPU. You cannot use wildcards in the file specification.

- If you use /COMMAND without specifying a file, VAXTPU tries to read a file called TPU$COMMAND.TPU from your current (default) directory.
- To specify a particular command file, define the logical name TPU$COMMAND or use /COMMAND= and specify the file.
- If you have defined TPU$COMMAND and do not want to run a command file, use /NOCOMMAND. (This decreases startup time by eliminating the search for a command file.)
- If a command you specify is not found, the editing session is aborted and you are returned to the DCL level.

/CREATE (default)
/NOCREATE

Controls whether VAXTPU creates an editing buffer when the specified input file is not found. Processing this qualifier depends on the VAXTPU application you are using. To find out whether the qualifier was issued when VAXTPU was invoked, use the built-in procedure GET_INFO. For example:

```
X := GET_INFO (COMMAND_LINE, "CREATE")
```

(For information on GET_INFO, see the *VAX Text Processing Utility Manual*.)

By default, EVE creates a buffer in which to edit a file. If you write out the contents of the buffer (with the built-in procedure WRITE_FILE or the EVE command WRITE FILE, or by exiting from the editor), VAXTPU creates a new file or new version of an existing file. If you use /NOCREATE and specify an input file that does not exist, the editing session is aborted and you are returned to the DCL command level. For example:

```
$ EDIT/TPU/NOCREATE NEWFILE.TXT

Input file does not exist: DISK$:[USER]NEWFILE.TXT;
$
```

/DEBUG[=debugger-file]
/NODEBUG (default)

Determines whether a debugger is available during an editing session. Using /DEBUG, with or without specifying a debugger file name, makes a debugger available.

If you use the /DEBUG qualifier but do not specify a debugger file name, VAXTPU reads, compiles, and executes the VAXTPU debugger, SYS$SHARE:TPU$DEBUG.TPU. VAXTPU runs this file before it runs either TPU$INIT_PROCEDURE or the file specified with the /COMMAND qualifier.

The VAXTPU debugger has fifteen commands to manipulate variables and control program execution. To start editing the code in the file you are debugging, use the VAXTPU debugger command GO. For more information on the VAXTPU debugger, see the *VAX Text Processing Utility Manual.*

To call a debugger file other than SYS$SHARE:TPU$DEBUG.TPU, use the /DEBUG= qualifier and specify the device, directory, and file name. If you specify only the file name, VAXTPU searches SYS$SHARE for the file. You can also define the logical name TPU$DEBUG to specify a file containing a debugger program. Once you define this logical, using /DEBUG without specifying a file calls the specified file TPU$DEBUG.

/DISPLAY[=file-spec] (default)
/NODISPLAY

Determines whether a VAXTPU session is run from a supported terminal and uses terminal functions such as the screen display and the keyboard. By default, the session is run with the screen management file, TPU$CCTSHR.EXE, for terminals that respond to ANSI control functions and that operate in ANSI mode.

VAXTPU expects sessions to be run from a supported terminal unless you specify /NODISPLAY. If you use /DISPLAY with an unsupported input device, VAXTPU displays an error message and the session is terminated. For information on terminals supported by VAXTPU, see the *VAX Text Processing Utility Manual.*

The /NODISPLAY qualifier causes VAXTPU to run without using the screen display and the keyboard functions of a terminal. Typically, you use /NODISPLAY when running VAXTPU procedures in a batch job or when using VAXTPU on an unsupported terminal.

If you use /NODISPLAY, window and screen manipulation commands and key definitions cause errors. Screen manipulation statements (ADJUST_WINDOW, CREATE_WINDOW, MAP) and key definitions will run, but the statements are meaningless and may display error messages in the batch log file or on your screen. When using /NODISPLAY, you must use a special startup file (either a section file or a command file) for the session. This startup file should follow these rules:

- The file should not include statements for key definitions or screen manipulation, except for READ_LINE, MESSAGE, and LAST_KEY, which work with some restrictions. (For a list of the restrictions, see the descriptions of READ_LINE and LAST_KEY in the *VAX Text Processing Utility Manual.*)

- The file should be a complete VAXTPU session; it should end with EXIT or QUIT.

/INITIALIZATION[=initialization-file] /NOINITIALIZATION

Determines whether VAXTPU executes a user-written initialization file containing editor commands, typically to set editing values or private defaults (such as margins and tab stops). The default for this qualifier depends on the section file used during the editing session.

Initialization files differ from command files in that initialization files contain EVE commands, while command files contain VAXTPU procedures and statements. The default file type for initialization files is EVE. (For more information on writing and using initialization files, see the *VAX Text Processing Utility Manual.*)

With EVE, if you use /INITIALIZATION without specifying an initialization file, VAXTPU searches your current (default) directory and SYS$LOGIN for a file called EVE$INIT.EVE. To specify a particular initialization file, define the logical name EVE$INIT or use /INITIALIZATION= and specify the file you want. You cannot use wildcards in the file specification, and you cannot nest initialization files.

If you do not want to use any initialization file, use /NOINITIALIZATION.

Commands in the initialization file for buffer settings (such as margins and tabs) apply to the main buffer and to a system buffer called $DEFAULTS$. Buffers you create during the session will assume the same buffer settings as $DEFAULTS$ for margins, tab stops, direction, and mode. VAXTPU runs a section file first (and fastest), then a command file (if any), and then an initialization file. Thus, key definitions in an initialization file override those in a section file.

/JOURNAL[=journal-file] (default) /NOJOURNAL

Determines whether VAXTPU keeps a journal file of your editing session, so you can recover your work in case of a system failure. Processing this qualifier depends on the VAXTPU application you are using. To find out whether the qualifier was issued when VAXTPU was invoked, use the built-in procedure GET_INFO. For example:

```
X := GET_INFO (COMMAND_LINE, "JOURNAL")
```

(For information on GET_INFO, see the *VAX Text Processing Utility Manual.*)

VAXTPU applications create the journal file in the current (default) directory. By default, EVE maintains a journal file that has the same name as the input file and the file type TJL. If you invoke VAXTPU without an input file specification, the default name for the journal file is TPU.TJL. If you want the journal file to have a different name or to be created in a different directory, use /JOURNAL= and specify the journal file.

To prevent VAXTPU from keeping a journal file for your editing session, use the qualifier /NOJOURNAL.

To recover your edits, invoke VAXTPU by reissuing the command for the original, interrupted session (including all qualifiers) and adding /RECOVER.

/MODIFY
/NOMODIFY

Determines whether the first buffer in an editing session is modifiable. If you use /MODIFY, the first buffer is modifiable. If you use /NOMODIFY, the first buffer is unmodifiable. If you do not specify either form of the qualifier, VAXTPU determines the buffer status from the /READ_ONLY or /WRITE qualifier. By default, a read-only buffer is unmodifiable and a write buffer is modifiable.

With EVE, the /NO]MODIFY qualifier overrides /[NO]READ_ONLY and /[NO]WRITE; that is, if you use inconsistent combinations of /[NO]MODIFY with /[NO]READ_ONLY or /[NO]WRITE, the buffer status is determined according to /MODIFY or /NOMODIFY.

/OUTPUT=output-file (default)
/NOOUTPUT

Determines the specification of the output file that is created at the end of your editing session. Processing this qualifier depends on the VAXTPU application you are using. To find out whether the qualifier was issued when VAXTPU was invoked, use the built-in procedure GET_INFO. For example:

```
X := GET_INFO (COMMAND_LINE, "OUTPUT")
```

(For information on GET_INFO, see the *VAX Text Processing Utility Manual.*)

By default, EVE uses the same file specification for both the input file and the output file. The output file has a version number one higher than the highest existing version of the input file. To specify an output file different from the input file (such as to write the output in a diferent directory or device), use /OUTPUT= and specify the output file you want.

Using /NOOUTPUT suppresses the creation of an output file, but does not affect creation of a journal file. If you use /NOOUTPUT, EVE makes the main or first buffer in an edit session a no-write buffer. If you enter /NOOUTPUT and then decide you want an output file, use the EVE command WRITE FILE before ending the editing session to write out the contents of a buffer.

/READ_ONLY
/NOREAD_ONLY (default)

Determines whether VAXTPU creates an output file from the contents of the main buffer. With /NOREAD_ONLY, VAXTPU creates an output file from the contents of the main buffer if you modified it.

Using the qualifier /READ_ONLY is effectively the same as using /NOOUTPUT for the main buffer. When you specify /READ_ONLY, VAXTPU sets the NO_WRITE and NO_MODIFY attributes for the main buffer. When a buffer is set to NO_WRITE, the contents of the buffer are not written out when you exit from VAXTPU. For example, both the EXIT and QUIT commands end the editing session without creating a new file from the contents of the main buffer (even if you modified it).

Typically, you use /READ_ONLY for demonstration sessions or to examine a file without making any edits. (If you do make edits and want to save them, use the EVE command WRITE FILE before ending the session.)

You cannot specify the combination of /READ_ONLY and /WRITE, or the combination of /NOREAD_ONLY and /NO_WRITE on the same command line. VAXTPU signals an error and returns control to DCL if it encounters either of these combinations.

/RECOVER
/NORECOVER (default)

Determines whether VAXTPU reads a journal file at the start of an editing session to recover your work from an interrupted session.

To recover your edits, reenter the command for the original, interrupted session, including all qualifiers, and add /RECOVER. VAXTPU then processes the journal file and recovers your work. (The last few keystrokes may have been lost.) If you originally used /JOURNAL= and specified a journal file other than the default (for example, if you specified a different directory for the journal file or a file type other than TJL), you must also use /JOURNAL= and specify the journal file.

When you recover a session, all files must be in the same state as at the start of the editing session being recovered. All terminal characteristics must also be in the same state as at the start of that session. Carefully check the following terminal characteristics:

Device_Type
Edit_mode
Eightbit
Page
Width

/SECTION[=file-spec] (default)
/NOSECTION

Determines whether VAXTPU reads a section file, containing in binary form, key definitions and compiled procedures. The default file type for section files is TPU$SECTION. By default, VAXTPU tries to read the file SYS$SHARE:TPU$SECTION.TPU$SECTION, which is the EVE section file.

To specify a different section file, define TPU$SECTION or use /SECTION= and specify the section file you want. Use a full file specification for the section file, including the device (or disk) and directory. Otherwise, VAXTPU assumes it is in SYS$SHARE. To create a section file, use the built-in procedure SAVE or the EVE command SAVE EXTENDED TPU.

If you do not define TPU$SECTION or do not specify a section file with /SECTION= on the command line, VAXTPU defines TPU$SECTION to point to SYS$SHARE:EVE$SECTION.TPU$SECTION.

If you specify /NOSECTION, VAXTPU does not read a section file. If no section file is read, and no command file is read, VAXTPU will not have a user interface and no keys will be defined. In this state, the only way to exit from VAXTPU is to press CTRL/Y. Typically, you use /NOSECTION when creating your own VAXTPU application without using EVE as a base.

/START_POSITION=(row, column) (default= 1,1)

Determines the row and column in the main buffer where the cursor first appears. By default, the start position is the first row, first column (upper left corner) of the buffer. Typically, you use /START_POSITION when you want to begin editing at a known or particular line or column.

/WRITE(default)
/NOWRITE

Determines whether you can edit text in the first buffer in a session. If you specify /NOWRITE, VAXTPU creates a read-only buffer for the first input file. VAXTPU does not create a new output file for such a buffer. This qualifier does not affect buffers other than the first buffer created during the session.

If you specify /WRITE or if you do not specify any qualifier related to /READ_ONLY or /WRITE, the first buffer is a write buffer. The /WRITE qualifier is the opposite of the /READ_ONLY qualifier.

You cannot specify the combination of /WRITE and /READ_ONLY, or the combination and /NO_WRITE and /NOREAD_ONLY on the same command line. VAXTPU signals an error and returns control to DCL if it encounters either of these combinations.

EXAMPLES

1 `$ EDIT/TPU/OUTPUT=newfile.txt oldfile.txt`

The EDIT/TPU command in this example invokes VAXTPU (running EVE) to edit the file OLDFILE.TXT. Modifying the main buffer and then using the command EXIT to end the session creates an output file called NEWFILE.TXT.

2 `$ EDIT/TPU/SECTION=disk$1:[user]vt100ini`

The EDIT/TPU command in this example invokes VAXTPU, using the section file VT100INI.TPU$SECTION instead of the EVE section file.

3 `$ EDIT/TPU/RECOVER oldfile.txt`

The EDIT/TPU command in this example invokes VAXTPU (running EVE) to recover your work from an interrupted session. VAXTPU opens the file OLDFILE.TXT and then processes the journal file OLDFILE.TJL. Once the journal file has been processed, you can resume interactive editing.

4 `$ EDIT/TPU/RECOVER/JOURNAL=save.xxx memo.dat`

The EDIT/TPU command in this example invokes VAXTPU (running EVE) to recover your edits from an interrupted session. VAXTPU opens the file MEMO.DAT and then processes the journal file SAVE.XXX.

ENDSUBROUTINE

Defines the end of a subroutine in a command procedure. For more information about the ENDSUBROUTINE command, refer to the description of the CALL command

FORMAT ENDSUBROUTINE

EOD

Signals the end of a data stream when a command or program is reading data from an input device other than an interactive terminal. Used to end a data line that begins with a dollar sign. Also, used to end an input file if more than one input file is contained in the command stream without intervening commands.

FORMAT

$ EOD

PARAMETERS

None.

DESCRIPTION

The EOD (End of Deck) command in a command procedure or in a batch job does the following:

- Terminates input data lines that begin with dollar signs. The DECK command indicates that the following lines begin with dollar signs and should be interpreted as data, not as commands; the EOD command indicates the end of the data lines.
- Terminates an input file if multiple input files are contained in the command stream without intervening commands. The program or command reading the data receives an end-of-file condition when the EOD command is read.

The EOD command must be preceded by a dollar sign; the dollar sign must be in the first character position (column 1) of the input record.

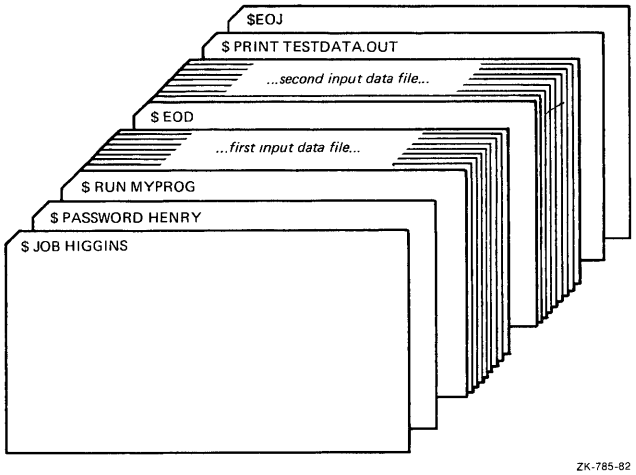
EXAMPLES

1

```
$ CREATE WEATHER.COM
$ DECK
$ FORTRAN WEATHER
$ LINK WEATHER
$ RUN WEATHER
$ EOD
$ @WEATHER
```

In this example, the command procedure creates a command procedure called WEATHER.COM. The lines delimited by the DECK and EOD commands are written to the file WEATHER.COM. Then the command procedure executes WEATHER.COM.

2



The program MYPROG requires two input files; these are read from the logical device SYS$INPUT. The EOD command signals the end of the first data file and the beginning of the second. The next line that begins with a dollar sign (a PRINT command in this example) signals the end of the second data file.

EOJ

Marks the end of a batch job submitted through a card reader. An EOJ card is not required; however, if present, the first nonblank character in the command line must be a dollar sign ($). If issued in any other context, the EOJ command logs the process out. The EOJ command cannot be abbreviated.

The EOF card is equivalent to the EOJ card.

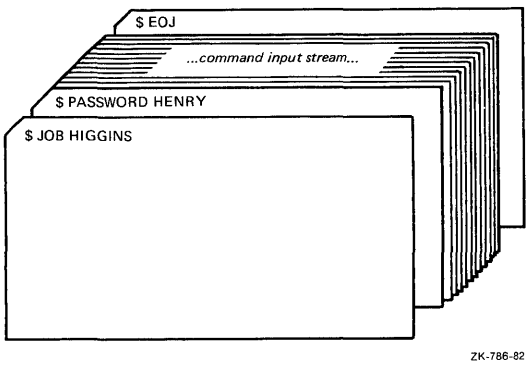
FORMAT

$ EOJ

PARAMETERS

None.

EXAMPLE



The JOB and PASSWORD commands mark the beginning of a batch job submitted through the card reader; the EOJ command marks the end of the job.

EXAMINE

Displays the contents of virtual memory.

Requires user mode read (R) and write (W) access to the virtual memory location whose contents you want to examine.

FORMAT

EXAMINE *location[:location]*

PARAMETER

location[:location]

Specifies a virtual address or a range of virtual addresses (where the second address is larger than the first) whose contents you want to examine. If you specify a range of addresses, separate the first and last with a colon.

A location can be any valid arithmetic expression containing arithmetic or logical operators or previously assigned symbols. Radix qualifiers determine the radix in which the address is interpreted; hexadecimal is the initial default radix. Symbol names are always interpreted in the radix in which they were defined. The radix operators %X, %D, or %O can precede the location. A hexadecimal value must begin with a number (or be preceded by %X).

The DEPOSIT and EXAMINE commands maintain a pointer to the current memory location. The EXAMINE command sets this pointer to the last location examined when you specify an EXAMINE command. You can refer to this location using the period (.) in a subsequent EXAMINE or DEPOSIT command.

DESCRIPTION

When the EXAMINE command is executed, it displays the virtual memory address in hexadecimal format and the contents in the radix requested as follows:

```
address:  contents
```

If the address specified is not accessible to user mode, four asterisks are displayed in the contents field.

Radix Qualifiers: The radix default for a DEPOSIT or EXAMINE command determines how the command interprets numeric literals. The initial default radix is hexadecimal; all numeric literals in the command line are assumed to be hexadecimal values. If a radix qualifier modifies an EXAMINE command, that radix becomes the default for subsequent EXAMINE and DEPOSIT commands, until another qualifier overrides it. For example:

```
$ EXAMINE/DECIMAL 900
00000384:  0554389621
```

The EXAMINE command interprets the location 900 as a decimal number and displays the contents of that location in decimal. All subsequent DEPOSIT and EXAMINE commands assume that numbers you enter for addresses and data are decimal. Note that the EXAMINE command always displays the address location in hexadecimal format.

Symbol names defined by = (Assignment Statement) commands are always interpreted in the radix in which they were defined.

Note that hexadecimal values entered as examine locations or as data to be deposited must begin with a numeric character (0 through 9). Otherwise, the command interpreter assumes that you have entered a symbol name, and attempts symbol substitution.

You can use the radix operators %X, %D, or %O to override the current default when you enter the EXAMINE command. For example:

```
$ EXAMINE/DECIMAL %X900
00000900:  321446536
```

This command requests a decimal display of the data in the location specified as hexadecimal 900.

Length Qualifiers: The initial default length unit for the EXAMINE command is a longword. The EXAMINE command displays data, one longword at a time, with blanks between longwords. If a length qualifier modifies the command, that length becomes the default length of a memory location for subsequent EXAMINE and DEPOSIT commands, until another qualifier overrides it.

Restriction on Placement of Qualifiers: The EXAMINE command analyzes expressions arithmetically. Therefore, qualifiers are interpreted correctly only when they appear immediately after the command name.

QUALIFIERS

/ASCII
Requests that the data at the specified location be displayed in ASCII.

Binary values that do not have ASCII equivalents are displayed as periods (.).

When you specify /ASCII, or when ASCII mode is the default, hexadecimal is used as the default radix for numeric literals that are specified on the command line.

/BYTE
Requests that data at the specified location be displayed one byte at a time.

/DECIMAL
Requests that the contents of the specified location be displayed in decimal format.

/HEXADECIMAL
Requests that the contents of the specified location be displayed in hexadecimal format.

/LONGWORD
Requests that data at the specified location be displayed one longword at a time.

/OCTAL
Requests that the contents of the specified location be displayed in octal format.

/WORD
Requests that data at the specified location be displayed one word at a time.

EXAMPLES

1

```
$ RUN   MYPROG
CTRL/Y
$ EXAMINE   2678
0002678:  1F4C5026
$ CONTINUE
```

In this example, the RUN command begins execution of the image MYPROG.EXE. While MYPROG is running, CTRL/Y interrupts its execution, and the EXAMINE command requests a display of the contents of virtual memory location 2678 (hexadecimal).

2

```
$ BASE = %X1C00
$ READBUF = BASE + %X50
$ ENDBUF = BASE + %XA0
$ RUN   TEST
CTRL/Y
$ EXAMINE/ASCII READBUF:ENDBUF
00001C50:  BEGINNING OF FILE MAPPED TO GLOBAL SECTION
    .
    .
    .
```

In this example, before executing the program TEST.EXE, symbolic names are defined for the program's base address and for labels READBUF and ENDBUF; all are expressed in hexadecimal format using the radix operator %X. READBUF and ENDBUF define offsets from the program base.

While the program is executing, CTRL/Y interrupts it, and the EXAMINE command requests a display in ASCII of all data between the specified memory locations.

EXCHANGE

Invokes the Exchange Utility (EXCHANGE) to manipulate mass storage volumes that are written in formats other than those normally recognized by the VMS operating system.

EXCHANGE allows you to perform any of the following tasks:

- Create foreign volumes
- Transfer files to and from the volume
- List directories of the volume

For block-addressable devices, such as RT–11 disks, EXCHANGE performs additional operations such as renaming and deleting files. The Exchange Utility can also manipulate Files–11 files that are images of foreign volumes; these files are called *virtual devices*.

The /[NO]MESSAGE qualifier determines whether EXCHANGE displays information related to EXCHANGE INITIALIZE, MOUNT, and DISMOUNT subcommands. You can also use this qualifier with any of these three subcommands to reverse the default. Normally, EXCHANGE displays the information.

For a complete description of the Exchange Utility, see the *VMS Exchange Utility Manual*.

FORMAT

EXCHANGE *[subcommand] [file-spec] [file-spec]*

EXIT

Terminates processing of a command procedure and returns control to the next higher command level — either an invoking command procedure or DCL. The EXIT command also terminates an image normally after a user enters CTRL/Y (executing another image has the same effect).

FORMAT

EXIT *[status-code]*

PARAMETER

status-code

Defines a numeric value for the reserved global symbol $STATUS. You can specify the status-code as an integer or an expression equivalent to an integer value. The value can be tested by the next outer command level. The low-order three bits of the value determine the value of the global symbol $SEVERITY.

If you specify a status-code, DCL interprets the code as a condition code. Note that even numeric values produce warning, error, and fatal error messages, and that odd numeric values produce either no message or a success or informational message.

If you do not specify a status-code, the current value of $STATUS is saved. When control returns to the outer command level, $STATUS contains the status of the most recently executed command or program.

DESCRIPTION

The EXIT and STOP commands both provide a way to terminate the execution of a procedure. The EXIT command terminates execution of the current command procedure and returns control to the calling command level. If you enter the EXIT command from a noninteractive process (such as a batch job), at command level 0, then the process terminates.

The STOP command returns control to command level 0, regardless of the current command level. If you execute the STOP command from a command procedure or from a noninteractive process (such as a batch job) the process terminates.

When a DCL command, user program, or command procedure completes execution, the command interpreter saves the condition code value in the global symbol $STATUS. The system maintains this value in hexadecimal. If an EXIT command does not explicitly set a value for $STATUS, the command interpreter uses the current value of $STATUS to determine the error status.

The low-order three bits of the status value contained in $STATUS represent the severity of the condition. The reserved global symbol $SEVERITY contains this portion of the condition code. Severity values range from zero through four, as follows:

Value	Severity
0	Warning
1	Success
2	Error
3	Information
4	Severe (fatal) error

Note that the success and information codes have odd numeric values, and that warning and error codes have even numeric values.

When any command procedure exits and returns control to another level, the command interpreter tests the current value of $STATUS. If $STATUS contains an even numeric value and if its high-order digit is 0, the command interpreter will display the system message associated with that status code, if one exists. (If no message exists, the message NOMSG will be displayed.) If the high order digit is 1, the message is not displayed.

When a command procedure exits following a warning or error condition that has already been displayed by a DCL command, the command interpreter sets the high-order digit of $STATUS to 1, leaving the remainder of the value intact. This ensures that error messages are not displayed by both the command that caused the error, and by the command procedure.

The EXIT command, when used after you interrupt an image with CTRL/Y, causes a normal termination of the image that is currently executing. If the image declared any exit-handling routines, they are given control. This is in contrast to the STOP command, which does not execute exit-handling routines. For this reason, the EXIT command is generally preferable to the STOP command.

EXAMPLES

1
```
$ EXIT 1
```

The EXIT command in this example exits to the next higher command level giving $STATUS and $SEVERITY a value of 1.

2
```
$ ON WARNING THEN EXIT
$ FORTRAN 'P1'
$ LINK 'P1'
$ RUN 'P1'
```

The EXIT command in this example is used as the target of an ON command; this statement ensures that the command procedure terminates whenever any warnings or errors are issued by any command in the procedure.

The procedure exits with the status value of the command or program that caused the termination.

3

```
$ START:
$       IF (P1 .EQS. "TAPE") .OR. (P1 .EQS. "DISK") THEN GOTO 'P1'
$       INQUIRE P1 "Enter device (TAPE or DISK)"
$       GOTO START
$ TAPE: !  Process tape files
   .
   .
   .
$       EXIT
$ DISK:  ! Process disk files
   .
   .
   .
$       EXIT
```

The command procedure in this example shows how to use the EXIT command to terminate different command paths within the procedure. To execute the procedure, you must enter either TAPE or DISK as a parameter. The IF command uses a logical OR to test whether either of these strings was entered. If the result is true, the GOTO command branches to the corresponding label. If P1 was neither TAPE nor DISK, the INQUIRE command prompts for a correct parameter.

The commands following each of the labels TAPE and DISK provide different paths through the procedure. The EXIT command before the label DISK ensures that the commands after the label DISK are executed only if the procedure explicitly branches to DISK.

Note that the EXIT command at the end of the procedure is not required because the end of the procedure causes an implicit EXIT command. Use of the EXIT command, however, is recommended.

4

```
$ IF P1. EQS. "" THEN -
     INQUIRE P1 "Enter file-spec (null to exit)"
$ IF P1 .EQS. "" THEN EXIT
$ PRINT 'P1'/AFTER=20:00/COPIES=50/FORMS=6
```

The command procedure in this example tests whether a parameter was passed to it; if the parameter was not passed, the procedure prompts for the required parameter. Then it retests the parameter P1. If a null string, indicated by a carriage return for a line with no data, is entered, the procedure exits. Otherwise, it executes the PRINT command with the current value of P1 as the input parameter.

5

```
$ IF P1 .EQS. "" THEN INQUIRE P1 "Code"
$ CODE = %X'P1'
$ EXIT CODE
```

The command procedure in this example, E.COM, illustrates how to determine the system message, if any, associated with a hexadecimal system status code. The procedure requires a parameter and prompts if none is entered. Then it prefixes the value with the radix operator %X and assigns this string to the symbol CODE. Finally, it issues the EXIT command with the hexadecimal value. For example, if the procedure is in the file E.COM:

```
$ @E 1C
%SYSTEM-F-EXQUOTA, exceeded quota
```

When the procedure exits, the value of $STATUS is %X1C, which equates to the EXQUOTA message. Note that you can also use the F$MESSAGE lexical function to determine the message that corresponds to a status code.

```
6  $ RUN MYPROG
   CTRL/Y
   $ EXIT
```

In this example, the RUN command initiates execution of the image MYPROG.EXE. Then CTRL/Y interrupts the execution. The EXIT command that follows calls any exit handlers declared by the image before terminating MYPROG.EXE.

GOSUB

Transfers control to a labeled subroutine in a command procedure without creating a new procedure level.

FORMAT

GOSUB *label*

PARAMETER

label

Specifies a 1- through 255-alphanumeric character label appearing as the first item on a command line. A label may not contain embedded blanks. When the GOSUB command is executed, control passes to the command following the specified label.

The label can precede or follow the GOSUB statement in the current command procedure. When you use a label in a command procedure, it must be terminated with a colon. If you use duplicate labels, control is always given to the label most recently read by DCL.

DESCRIPTION

Use the GOSUB command in command procedures to transfer control to a subroutine specified by the label. If the command stream is not being read from a random access device (that is, a disk device), the GOSUB command performs no operation.

The RETURN command terminates the GOSUB subroutine procedure, returning control to the command following the calling GOSUB statement. The RETURN command accepts an optional status value.

The GOSUB command does not cause the creation of a new procedure level. Therefore, it is referred to as a "local" subroutine call. Any labels and local symbols defined in the current command procedure level are available to a subroutine invoked with a GOSUB command. The GOSUB command can be nested up to a maximum of 16 levels per procedure level.

When the command interpreter encounters a label, it enters the label in a label table. This table is allocated from space available in the local symbol table. If the command interpreter encounters a label that already exists in the table, the new definition replaces the existing one. Therefore, if you use duplicate labels, control is always given to the label most recently read by DCL. The following rules apply:

- If duplicate labels precede and follow the GOSUB command, control is given to the label preceding the command.
- If duplicate labels all precede the GOSUB command, control is given to the most recent label, that is, the one nearest the GOSUB command.
- If duplicate labels all follow the GOSUB command, control is given to the one nearest the GOSUB command.

If a label does not exist in the current command procedure, the procedure cannot continue and is forced to exit.

Note that the amount of space available for labels is limited. If a command procedure uses many symbols and contains many labels, the command interpreter may run out of table space and issue an error message.

EXAMPLE

```
$!
$! GOSUB.COM
$!
$ SHOW TIME
$ GOSUB TEST1
$ WRITE SYS$OUTPUT "success completion"
$ EXIT
$!
$! TEST1 GOSUB definition
$!
$ TEST1:
$     WRITE SYS$OUTPUT "This is GOSUB level 1."
$     GOSUB TEST2
$     RETURN %X1
$!
$! TEST2 GOSUB definition
$!
$ TEST2:
$     WRITE SYS$OUTPUT "This is GOSUB level 2."
$     GOSUB TEST3
$     RETURN
$!
$! TEST3 GOSUB definition
$!
$ TEST3:
$     WRITE SYS$OUTPUT "This is GOSUB level 3."
$     RETURN
```

This sample command procedure shows how to use the GOSUB command to transfer control to labeled subroutines. The GOSUB command transfers control to the subroutine labeled TEST1. The procedure executes the commands in subroutine TEST1, branching to the subroutine labeled TEST2. The procedure then executes the commands in subroutine TEST2, branching to the subroutine labeled TEST3. Each subroutine is terminated by the RETURN command. After TEST3 is executed, the RETURN command returns control back to the command line following each calling GOSUB statement. At this point, the procedure has been successfully executed.

GOTO

Transfers control to a labeled statement in a command procedure.

FORMAT

GOTO *label*

PARAMETER

label

Specifies a 1- through 255-alphanumeric character label appearing as the first item on a command line. A label may not contain embedded blanks. When the GOTO command is executed, control passes to the command following the specified label.

The label can precede or follow the GOTO statement in the current command procedure. When you use a label in a command procedure, it must be terminated with a colon. If you use duplicate labels, control is always given to the label most recently read by DCL.

DESCRIPTION

Use the GOTO command in command procedures to transfer control to a line that is not the next line in the procedure. If the command stream is not being read from a random access device (that is, a disk device), the GOTO command performs no operation.

When the command interpreter encounters a label, it enters the label in a label table. This table is allocated from space available in the local symbol table. If the command interpreter encounters a label that already exists in the table, the new definition replaces the existing one. Therefore, if you use duplicate labels, control is always given to the label most recently read by DCL. In general:

- If duplicate labels precede and follow the GOTO command, control is given to the label preceding the command.
- If duplicate labels all precede the GOTO command, control is given to the most recent label, that is, the one nearest the GOTO command.
- If duplicate labels all follow the GOTO command, control is given to the one nearest the GOTO command.

If a label does not exist in the current command procedure, the procedure cannot continue and is forced to exit.

Note that the amount of space available for labels is limited. If a command procedure uses many symbols and contains many labels, the command interpreter may run out of table space and issue an error message.

EXAMPLES

1

```
$ IF P1 .EQS. "HELP" THEN GOTO TELL
$ IF P1 .EQS. "" THEN GOTO TELL
   .
   .
   .
$ EXIT
$ TELL:
$ TYPE SYS$INPUT
To use this procedure, you must enter a value for P1.
   .
   .
   .
$ EXIT
```

In this example, the IF command checks the first parameter passed to the command procedure; if this parameter is the string HELP or if the parameter is not specified, the GOTO command is executed and control is passed to the line labeled TELL. Otherwise, the procedure continues executing until the EXIT command is encountered. At the label TELL, a TYPE command displays data in the input stream that documents how to use the procedure.

2

```
$ ON ERROR THEN GOTO CHECK
   .
   .
   .
$ EXIT
$ CHECK:  ! Error handling routine
   .
   .
   .
$ END:
$ EXIT
```

The ON command establishes an error-handling routine. If any command or procedure subsequently executed in the command procedure returns an error or severe error return, the GOTO command transfers control to the label CHECK.

HELP

Displays information concerning use of the system, including formats and explanations of commands, parameters, and qualifiers.

FORMAT

HELP *[keyword ...]*

PARAMETER

keyword ...

Specifies one or more keywords that refer to the topic or subtopic on which you want information from a HELP library. To use the HELP facility in its simplest form, enter the HELP command from your terminal. The HELP facility displays a list of topics at your terminal and the prompt Topic?. To see information on one of the topics, type the topic name after the prompt. The system displays information on that topic.

If the topic has subtopics, the HELP command lists the subtopics and displays the Subtopic? prompt. To get information on one of the subtopics, type the name after the prompt. To see information on another topic, press RETURN. You can now ask for information on another topic when HELP displays the Topic? prompt. Press RETURN to exit the HELP facility and return to the DCL command level.

DESCRIPTION

Information within HELP libraries is arranged in a hierarchical manner. The levels are as follows:

1. None—If you do not specify a keyword, HELP describes the HELP command and lists the topics that are documented in the root library. Each item in the list is a keyword in the first level of the hierarchy.
2. Topic-name—If you specify a keyword by naming a topic, HELP describes the topic as it is documented in either the root library or one of the other enabled default libraries. Keywords for additional information available on this topic are listed.
3. Topic-name subtopic—If you specify a subtopic following a topic, HELP provides a description of the specified subtopic.
4. @file-spec followed by any of the above—If you specify a HELP library to replace the current root library, HELP searches that library for a description of the topic or subtopic specified. The file specification must take the same form as the file specification included with the /LIBRARY command qualifier. However, if the specified library is an enabled user-defined default library, the file specification can be abbreviated to any unique leading substring of that default library's logical name translation.

If you use an asterisk in place of any keyword, the HELP command displays all information available at the level that the asterisk replaces. For example, HELP COPY * displays all the subtopics under the topic COPY.

If you use an ellipsis immediately after any primary keyword, HELP displays all the information on the specified topic and all subtopics of that topic. For example, HELP COPY... displays information on the COPY topic as well as information on all the subtopics under COPY. The ellipsis can only be used from the topic level; it cannot be used from the subtopic level.

You can use percent signs and asterisks in the keyword as wildcard characters.

QUALIFIERS

/INSTRUCTIONS (default)
/NOINSTRUCTIONS

Displays an explanation of the HELP command along with the list of topics (if no topic is specified). By default, the HELP command display includes a description of the facility and the format, along with the list of topics. If you specify /NOINSTRUCTIONS, only the list of topics is displayed.

/LIBLIST (default)
/NOLIBLIST

Displays any auxiliary help libraries.

/LIBRARY=file-spec
/NOLIBRARY

Uses an alternate HELP library instead of the default system library, SYS$HELP:HELPLIB.HLB. The specified library is used as the main (root) HELP library, and is searched for HELP information before any user-defined default HELP libraries are checked.

If you omit the device and directory specification, the default is SYS$HELP, the logical name of the location of the system HELP libraries. The default file type is HLB.

/NOLIBRARY excludes the default HELP library from the library search order.

/OUTPUT[=file-spec]
/NOOUTPUT

Controls where the output of the command is sent. By default, the output is sent to SYS$OUTPUT, the current process default output stream or device.

If you enter /OUTPUT with a partial file specification (for example, /OUTPUT=[JONES]), HELP is the default file name and LIS the default file type. No wildcards are allowed.

If you enter /NOOUTPUT, output is suppressed.

/PAGE (default)
/NOPAGE

Stops the display when the screen is full. You must press RETURN to continue.

If you specify /NOPAGE, output continues until the information display ends or until you manually control the scrolling.

/PROMPT (default)
/NOPROMPT

Permits you to solicit further information interactively. If you specify /NOPROMPT, HELP returns you to the DCL command level after displaying the requested information.

If /PROMPT is in effect, one of four different prompts is displayed, requesting you to specify a particular HELP topic or subtopic. Each prompt represents a different level in the hierarchy of HELP information. The four prompt levels are as follows:

1. Topic?—The root library is the main library and you are not currently examining HELP for a particular topic.
2. [library-spec] Topic?—The root library is a library other than the main library and you are not currently examining HELP for a particular topic.
3. [keyword] Subtopic?—The root library is the main library and you are currently examining HELP for a particular topic (or subtopic).
4. A combination of 2 and 3.

When you encounter one of these prompts, you can enter any one of the responses described in the following table:

Response	Action in the Current Prompt Environment
keyword[...]	(1,2) Searches all enabled libraries for the keyword. (3,4) Searches additional HELP libraries for the current topic (and/or subtopic) for the keyword.
@file-spec keyword[...]	(1,2) Same as above, except that the library specified by @file-spec is now the root library. If the specified library does not exist, HELP treats @file-spec as a normal keyword. (3,4) Same as above, treats @file-spec as a normal keyword. (1,2) Displays a list of topics available in the root library. (3,4) Displays the list of subtopics of the current topic (and/or subtopics) for which HELP exists.
RET	(1) Exits from HELP. (2) Changes root library to main library. (3,4) Prompts for a topic or subtopic at the next higher level.
CTRL/Z	(1,2,3,4) Exits from HELP.

/USERLIBRARY=(level[,...])
/NOUSERLIBRARY

Names the levels of search for information in auxiliary libraries.

PROCESS	Libraries defined at process level
GROUP	Libraries defined at group level
SYSTEM	Libraries defined at system level
ALL	All libraries (default)
NONE	No libraries (same as /NOUSERLIBRARY)

Auxiliary help libraries are libraries defined with the logical names HLP$LIBRARY, HLP$LIBRARY_1, HLP$LIBRARY_2, and so on. Libraries

are searched for information in this order: root (current) library, main library (if not current), libraries defined at process level, libraries defined at group level, libraries defined at system level, and the root library. If the search fails, the root library is searched a second time so that the context is returned to the root library from which the search was initiated. The default is /USERLIBRARY=ALL. If you specify only one level for HELP to search, you can omit the parentheses.

EXAMPLES

1

```
$ HELP
HELP
  .
  .  (HELP message text and list of topics)
  .
Topic?
```

In this example, the HELP command is entered without any qualifiers or parameters. This produces a display of the HELP topics available from the root HELP library, SYS$HELP:HELPLIB.HLB.

If you type one of the listed topics in response to the Topic? prompt, HELP displays information about that topic and a list of subtopics (if there are any). If one or more subtopics exist, HELP prompts you for a subtopic.

```
Topic? ASSIGN
ASSIGN
  .
  .  (HELP message text and subtopics)
  .
ASSIGN Subtopic?
```

If you type a subtopic name, HELP displays information about that subtopic:

```
ASSIGN Subtopic? Name
ASSIGN
  Name
  .
  .  (HELP message text and subtopics, if any)
  .
ASSIGN Subtopic?
```

If one or more sub-subtopics exist, HELP prompts for a sub-subtopic; otherwise, as in the previous example, the facility prompts you for another subtopic of the topic you are currently inspecting.

Typing a question mark redisplays the HELP message and options at your current level. Pressing RETURN does either of the following: (1) move you back to the previous HELP level if you are in a subtopic level, or (2) terminate HELP if you are at the first level. Pressing CTRL/Z terminates HELP at any level.

2

```
$ HELP COPY...
```

The HELP command in this example displays a description of the COPY command and of the command's parameters and qualifiers. Note that the ellipsis can only be used from the topic level; it cannot be used from the subtopic level.

```
3  $ HELP/NOPROMPT ASSIGN/GROUP
     .
     .  (ASSIGN/GROUP HELP message)
     .
   $
   $ HELP/NOPROMPT/PAGE EDIT *
     .
     .  (HELP messages on all first-level EDIT subtopics)
     .
   $
```

The two HELP commands request HELP on specific topics. In each case, HELP displays the HELP message you request and then returns you to DCL command level and the dollar sign prompt.

The first command requests HELP on the /GROUP qualifier of the ASSIGN command. The asterisk in the second example is a wildcard character. It signals HELP to display information about all EDIT subtopics, which HELP then displays in alphabetical order. The /NOPROMPT qualifier suppresses prompting in both sample commands. The /PAGE qualifier on the second HELP command causes output to the screen to stop after each screen of information is displayed.

```
4  $ HELP FILL
   Sorry, no documentation on FILL
   Additional information available:
    .
    .  (list of first-level topics )
    .
   Topic? @EDTHELP FILL
   FILL
    .
    .  (FILL HELP message)
    .
   @EDTHELP Topic?
```

When you enter a request for HELP on a topic that is not in the default HELP library, you can instruct HELP to search another HELP library for the topic. In this example, entering the command @EDTHELP FILL instructs HELP to search the HELP library SYS$HELP:EDTHELP.HLB for information on FILL, an EDT editor command. HELP displays the message and prompts you for another EDT editor topic.

```
5  $ DEFINE HLP$LIBRARY EDTHELP
   $ DEFINE HLP$LIBRARY_1 MAILHELP
   $ DEFINE HLP$LIBRARY_2 BASIC
   $ DEFINE HLP$LIBRARY_3 DISK2:[MALCOLM]FLIP
   $ HELP REM
```

You can use logical names to define libraries for HELP to search automatically if it does not find the specified topic in the VMS root HELP library. This sequence of commands instructs HELP to search libraries besides the default root library, SYS$HELP:HELPLIB.HLB.

The four DEFINE statements create logical names for the four user-defined HELP libraries that HELP is to search after it has searched the root library. The first three entries are HELP libraries in the directory. HELP searches by default for user-defined HELP libraries, SYS$HELP. The fourth is the HELP library FLIP.HLB in the directory DISK2:[MALCOLM]. Note that the logical names that you use to define these HELP libraries must be numbered consecutively; that is, you cannot skip any numbers.

HELP first searches the root library for REM. It then searches the libraries HLP$LIBRARY, HLP$LIBRARY_1, HLP$LIBRARY_2, and so on, until it finds REM or exhausts the libraries it knows it can search. When it finds REM in the BASIC.HLB library, it displays the appropriate HELP information and prompts you for a subtopic in that library. If you request HELP on a topic not in the BASIC.HLB library, HELP once again searches the HELP libraries you have defined.

IF

Tests the value of an expression and, depending on the syntax specified, executes

- one command following the THEN keyword if the expression is true
- multiple commands following the $THEN command if the expression is true
- one or more commands following the $ELSE command if the expression is false

FORMAT

$ IF *expression THEN [$] command*

or

$ IF *expression*
$ THEN *[command]*
command
.
.
.
$ [ELSE] *[command]*
command
.
.
.
$ ENDIF

PARAMETERS

expression

Defines the test to be performed. The expression can consist of one or more numeric constants, string literals, symbolic names, or lexical functions separated by logical, arithmetic, or string operators.

Expressions in IF commands are automatically evaluated during the execution of the command. Character strings beginning with alphabetic characters that are not enclosed in quotation marks are assumed to be symbol names or lexical functions. The Command Language Interpreter (CLI) replaces these strings with their current values.

Symbol substitution in expressions in IF commands is not iterative; that is, each symbol is replaced only once. However, if you want iterative substitution, precede a symbol name with an apostrophe or ampersand.

The command interpreter does not execute an IF command when it contains an undefined symbol. Instead, the command interpreter issues a warning message and executes the next command in the procedure.

For a summary of operators and details on how to specify expressions, see Chapter 6 of the *VMS DCL Concepts Manual*.

command

The DCL command or commands to be executed, depending on the syntax specified, when the result of the expression is true or false.

DESCRIPTION

The IF command tests the value of an expression and executes a given command if the result of the expression is true. The expression is true if the result has an odd integer value, a character string value that begins with the letters Y, y, T, or t, or an odd numeric string value.

The expression is false if the result has an even integer value, a character string value that begins with any letter except Y, y, T, or t, or an even numeric string value.

EXAMPLES

1

```
$ COUNT = 0
$ LOOP:
$ COUNT = COUNT + 1
    .
    .
    .
$ IF COUNT .LE. 10 THEN GOTO LOOP
$ EXIT
```

This example shows how to establish a loop in a command procedure, using a symbol named COUNT and an IF statement. The IF statement checks the value of COUNT and performs an EXIT command when the value of COUNT is greater than 10.

2

```
$ IF P1 .EQS. "" THEN GOTO DEFAULT
$ IF (P1 .EQS. "A") .OR. (P1 .EQS. "B") THEN GOTO 'P1'
$  WRITE SYS$OUTPUT "Unrecognized parameter option ''P1' "
$ EXIT
$  A:       !  Process option a
  .
  .
  .
$ EXIT
$  B:       !  Process option b
  .
  .
  .
$ EXIT
$  DEFAULT: !  Default processing
  .
  .
  .
$ EXIT
```

This example shows a command procedure that tests whether a parameter was passed. The GOTO command passes control to the label specified as the parameter.

If the procedure is executed with a parameter, the procedure uses that parameter to determine the label to branch to. For example:

```
@TESTCOM A
```

When the procedure executes, it determines that P1 is not null, and branches to the label A. Note that the EXIT command causes an exit from the procedure before the label B.

3
```
$  SET NOON
 .
 .
 .
$  LINK CYGNUS,DRACO,SERVICE/LIBRARY
$ IF $STATUS
$ THEN
$  RUN CYGNUS
$ ELSE
$   WRITE SYS$OUTPUT "LINK FAILED"
$ ENDIF
$ EXIT
```

This command procedure uses the SET NOON command to disable error checking by the command procedure. After the LINK command, the IF command tests the value of the reserved global symbol $STATUS. If the value of $STATUS indicates that the LINK command succeeded, then the program CYGNUS is run. If the LINK command returns an error status value, the command procedure issues a message and exits.

INITIALIZE

Formats a disk or magnetic tape volume and writes a label on the volume. At the end of initialization, the disk is empty except for the system files containing the structure information. All former contents of the disk are lost.

Requires VOLPRO privilege for most INITIALIZE operations.

FORMAT

INITIALIZE *device-name[:] volume-label*

PARAMETERS

device-name[:]

Specifies the name of the device on which the volume to be initialized is physically mounted.

The device does not have to be currently allocated; however, allocating the device before initializing it is the recommended practice.

volume-label

Specifies the identification to be encoded on the volume. For a disk volume, you can specify a maximum of 12 alphanumeric characters; for a magnetic tape volume, you can specify a maximum of 6 alphanumeric characters. Letters are automatically changed to uppercase. Nonalphanumeric characters are not allowed in the volume-label specification on disk.

To use ANSI "a" characters on the volume-label on magnetic tape, you must enclose the volume name in quotation marks. For an explanation of ANSI "a" characters, see the description of the /LABEL qualifier.

DESCRIPTION

The default format for disk volumes in the VMS operating system is called the Files–11 Structure Level 2. The default for magnetic tape volumes is based on Level 3 of the ANSI standard for magnetic tape labels and file structure for informational interchange (ANSI X3.27-1978).

The INITIALIZE command can also initialize disk volumes in the Files–11 Structure Level 1 format.

You do not need special privileges to override logical protection on the following devices:

- A blank disk or magnetic tape volume; that is, a volume that has never been written
- A disk volume that is owned by your current UIC or by the UIC [0,0]
- A magnetic tape volume that allows write access to your current UIC that was not protected when it was initialized

In all other cases, you must have the VOLPRO privilege to initialize a volume.

When the INITIALIZE command initializes a magnetic tape volume, it always attempts to read the volume. A blank magnetic tape can sometimes cause unrecoverable errors, such as the following:

- The message:

  ```
  %INIT-F-VOLINV, volume is invalid
  ```

- A runaway magnetic tape (this frequently occurs with new magnetic tapes that have never been written or that have been run through verifying machines). You can stop a runaway magnetic tape only by setting the magnetic tape drive off line and then putting it back on line.

If this type of unrecoverable error occurs, you can successfully initialize a magnetic tape by repeating the INITIALIZE command from an account that has the volume protection (VOLPRO) privilege and specifying the following qualifier in the command:

/OVERRIDE=(ACCESSIBILITY,EXPIRATION)

This qualifier ensures that the INITIALIZE command does not attempt to verify any labels on the magnetic tape.

Many of the INITIALIZE command qualifiers allow you to specify parameters that can maximize input/output efficiency.

QUALIFIERS

/ACCESSED=number-of-directories

Requires OPER privilege. Affects Files-11 Structure Level 1 disks ONLY.

Specifies, for disk volumes, the number of directories allowed in system space must be a value from 0 to 255. The default value is 3.

/BADBLOCKS=(area[,...])

Specifies, for disk volumes, faulty areas on the volume. The INITIALIZE command marks the areas as allocated so that no data is written in them.

Possible formats for area are as follows:

lbn[:count]	Logical block number of the first block and optionally a block count beginning with the first block, to be marked as allocated
sec.trk.cyl[:cnt]	Sector, track, and cylinder of the first block, and optionally a block count beginning with the first block, to be marked as allocated

All media supplied by DIGITAL and supported on the VMS operating system, except floppy disks and TU58 cartridges, are factory formatted and contain bad block data. The Bad Block Locator Utility (BAD) or the diagnostic formatter EVRAC can be used to refresh the bad block data or to construct it for the media exceptions above. The /BADBLOCKS qualifier is necessary only to enter bad blocks that are not identified in the volume's bad block data.

Digital Storage Architecture (DSA) disks (for example, disks attached to UDA-50 and HSC50 controllers) have bad blocks handled by the controller, and appear logically perfect to the file system.

For information on how to run BAD, see the *VMS Bad Block Locator Utility Manual*.

/CLUSTER_SIZE=number-of-blocks

Defines, for disk volumes, the minimum allocation unit, in blocks. The maximum size you can specify for a volume is one-hundredth the size of the volume; the minimum size you can specify is calculated with the following formula:

$$\frac{disk\ size(number\ of\ blocks)}{255 * 4096}$$

For Files–11 Structure Level 2 disks, the cluster size default depends on the disk capacity; disks that are 50,000 blocks or larger have a default cluster size of 3, while those smaller than 50,000 blocks have a default value of 1.

For Files–11 Structure Level 1 disks, the cluster size must always be 1.

/DATA_CHECK[=(option[,...])]

Checks all read and write operations on the disk. By default, no data checks are made. Specify one or both options:

READ	Checks all read operations
WRITE	Checks all write operations; default if only /DATA_CHECK is specified

To override the checking you specify at initialization for disks, enter a MOUNT command to mount the volume.

/DENSITY=density-value

The /DENSITY qualifier is not applicable to the TK50 tape device.

For floppy disk volumes that are to be initialized on RX02 dual-density disk drives, specifies the density at which the floppy disk is to be formatted.

For magnetic tape volumes, specifies the density in bytes per inch (bpi) at which the magnetic tape is to be written.

RX02 dual-density disk drives allow floppy disks to be initialized at single or double density. To specify single-density formatting of a floppy disk, specify the density value SINGLE. To specify double-density formatting of a floppy disk, specify the density value DOUBLE.

If you do not specify a density value for a floppy disk being initialized on an RX02 drive, the system leaves the volume at the density to which the volume was last formatted. Floppy disks purchased from DIGITAL are formatted in single density.

For magnetic tape volumes, the density value specified can be 800 bpi, 1600 bpi, or 6250 bpi, as long as the density is supported by the magnetic tape drive. If you do not specify a density value for a blank magnetic tape, the system uses a default density of the highest value allowed by the tape drive. If the drive allows 6250, 1600, and 800 bpi operation, the default density is 6250. If the drive allows only 1600 and 800 bpi operation then the default density is 1600. If you do not specify a density value for a magnetic tape that has been previously written, the system uses the density of the first record on the volume. The magnetic tape density will not default on an unusually short record.

Note: Floppy disks formatted in double density cannot be read or written by the console block storage device (an RX01 drive) of a VAX/780 until they have been reformatted in single density.

/DIRECTORIES=number-of-entries

Specifies, for disk volumes, the number of entries to preallocate for user directories. The number of entries must be an integer between 16 and 16000. The default value is 16.

/ERASE
/NOERASE (default)

Physically destroys deleted data (by writing over it). Controls the Data Security Erase (DSE) operation on the volume before initializing it. The /ERASE qualifier applies to ODS-2 disk and ANSI magnetic tape volumes, and is valid for magnetic tape devices that support the hardware erase function, such as TU78 and MSCP magnetic tapes.

If you specify /ERASE, a DSE operation is performed on the volume. For disk devices, the ERASE volume attribute is set. In effect, each file on the volume is erased when it is deleted.

Note that the amount of time taken by the DSE operation depends on the volume size; INITIALIZE/ERASE is always slower than INITIALIZE /NOERASE.

/EXTENSION=number-of-blocks

Affects Files-11 Structure Level 1 disks ONLY.

Specifies, for disk volumes, the number of blocks to use as a default extension size for all files on the volume. The extension default is used when a file increases to a size greater than its initial default allocation during an update. The value for the number-of-blocks parameter can range from 0 through 65,535. The default value is 5.

In VMS, the default file extension is specified using the SET RMS command.

/FILE_PROTECTION=code

Affects Files-11 Structure Level 1 disks ONLY.

Defines, for disk volumes, the default protection to be applied to all files on the volume.

Specify the code according to the standard syntax rules described in Section 8.1 in the *VMS DCL Concepts Manual*. Any attributes not specified are taken from the current default protection.

Note that this attribute is not used when the volume is being used on a VMS system, but is provided to control the process's use of the volume on RSX-11M systems. VMS systems always use the default file protection. Use the SET PROTECTION/DEFAULT command to change the default file protection.

/GROUP

Defines a group volume. The /GROUP qualifier applies protection of RWED to all ownership categories unless /GROUP is specified with /NOSHARE, in which case the volume protection is RWED for all but the world category. The owner UIC of the volume defaults to your group number and a member number of 0.

/HEADERS=number-of-headers

Specifies, for disk volumes, the number of file headers to be allocated for the index file. The minimum and default value is 16. The maximum is the value set with the /MAXIMUM_FILES qualifier.

/HIGHWATER (default)
/NOHIGHWATER

Affects Files-11 Structure Level 2 disks ONLY.

Sets the file highwater mark (FHM) volume attribute, which guarantees that a user cannot read data that he has not written. You cannot specify /NOHIGHWATER for magnetic tape.

The /NOHIGHWATER qualifier disables FHM for a disk volume.

/INDEX=position

Specifies the location of the index file for the volume's directory structure. Possible positions are as follows:

BEGINNING	Beginning of the volume
MIDDLE	Middle of the volume (default)
END	End of the volume
BLOCK:n	Beginning of the logical block specified by *n*

/LABEL=option

Defines characteristics for the magnetic tape volume label, as directed by the included option. The available options are as follows:

- OWNER_IDENTIFIER:"(14 ANSI characters)"

  Allows you to specify the Owner Identifier field in the volume label. The field specified can accept up to 14 ANSI characters.

- VOLUME_ACCESSIBILITY:"character"

  Specifies the character to be written in the volume accessibility field of the VMS ANSI volume label VOL1 on an ANSI magnetic tape. The character may be any valid ANSI "a" character. This set of characters includes numeric characters, uppercase letters, and any one of the following nonalphanumeric characters:

  ```
  ! " % ' ( ) * + , - . / : ; < = > ?
  ```

  By default, the VMS operating system provides a routine that checks this field in the following manner.

  - If the magnetic tape was created on a version of the VMS operating system that conforms to Version 3 of ANSI, then this option must be used to override any character other than an ASCII space.
  - If a VMS protection is specified and the magnetic tape conforms to an ANSI standard that is later than Version 3, then this option must be used to override any character other than an ASCII 1.

  If you specify any character other than the default, you must specify the /OVERRIDE=ACCESSIBILITY qualifier on the INITIALIZE and MOUNT commands in order to access the magnetic tape.

/MAXIMUM_FILES=n

Restricts the maximum number of files that the volume can contain. The /MAXIMUM_FILES qualifier overrides the default value, which is calculated as follows:

$$\frac{volume\ size\ in\ blocks}{(cluster\ factor + 1) * 2}$$

The maximum size you can specify for any volume is as follows:

$$\frac{volume\ size\ in\ blocks}{(cluster\ factor + 1)}$$

The minimum value is 0. Note that the maximum can be increased only by reinitializing the volume.

/OVERRIDE=(option[,...])

Requests the INITIALIZE command to ignore data on a magnetic tape volume that protects it from being overwritten. You may specify one or more of the following options:

ACCESSIBILITY	(For magnetic tapes only.) If the installation allows, this option overrides any character in the Accessibility Field of the volume. The necessity of this option is defined by the installation. That is, each installation has the option of specifying a routine that the magnetic tape file system will use to process this field. By default, VMS provides a routine that checks this field in the following manner. If the magnetic tape was created on a version of VMS that conforms to Version 3 of ANSI, this option must be used to override any character other than an ASCII space. If a VMS protection is specified and the magnetic tape conforms to an ANSI standard that is later than Version 3, this option must be used to override any character other than an ASCII 1. To use the ACCESSIBILITY option, you must have the user privilege VOLPRO or be the owner of the volume.
EXPIRATION	(For magnetic tapes only.) Allows you to write to a tape that has not yet reached its expiration date. You may need to do this for magnetic tapes that were created before VMS Version 4.0 on DIGITAL operating systems using the D% format in the volume Owner Identifier field. You must have the user privilege VOLPRO to override volume protection, or your UIC must match the UIC written on the volume.
OWNER_IDENTIFIER	Allows you to override the processing of the Owner Identifier field of the volume label.

If you specify only one option, you may omit the parentheses.

To initialize a volume that was initialized previously with the /PROTECTION qualifier, your UIC must match the UIC written on the volume or you must have VOLPRO privilege.

/OWNER_UIC=uic

Specifies an owner UIC for the volume. The default is your default UIC. Specify the UIC using standard UIC format as described in Section 8.1 in the *VMS DCL Concepts Manual.*

For magnetic tapes, no UIC is written unless protection on the magnetic tape is specified. If protection is specified, but no owner UIC is specified, your current UIC is assigned ownership of the volume.

/PROTECTION=(ownership[:access],...)

Applies the specified protection to the volume. Specify ownership as SYSTEM, OWNER, GROUP, or WORLD and access as R (read), W (write), E (execute), and D (delete). The default is your default protection. Note that the /GROUP, /SHARE, and /SYSTEM qualifiers can also be used to define protection for disk volumes.

For magnetic tape, the protection code is written to a VMS-specific volume label. The system only applies read and write access restrictions; execute and delete access are meaningless. Moreover, the system and the owner are always given both read and write access to magnetic tapes, regardless of the protection code you specify .

See Section 8.1 of the *VMS DCL Concepts Manual* for more information on specifying protection code. Any attributes not specified are taken from the current default protection.

When you specify a protection code for an entire disk volume, access type E (execute) indicates create access.

/SHARE (default)
/NOSHARE

Permits all categories of access by all categories of ownership. The /NOSHARE qualifier denies access to group (unless /GROUP is also specified) and world processes.

/STRUCTURE=level

Specifies whether the volume should be formatted in Files–11 Structure Level 1 or Structure Level 2 (the default). Level 1 is incompatible with the /DATA_CHECK and /CLUSTER_SIZE qualifiers. The default protection for a Structure Level 1 disk is full access to system, owner, and group, and R (read) access to all other users.

/SYSTEM

Requires a system UIC or SYSPRV privilege.

Defines a system volume. The owner UIC defaults to [1,1]. Protection defaults to complete access by all ownership categories, except that only system processes can create top-level directories.

/USER_NAME=name

Specifies a user name to be associated with the volume. The name must be 1 to 12 alphanumeric characters. The default is your user name.

/VERIFIED
/NOVERIFIED

Indicates whether the disk has bad block data on it. Use the /NOVERIFIED qualifier to ignore bad block data on the disk. The default is /VERIFIED for disks with 4096 blocks or more and /NOVERIFIED for disks with less than 4096 blocks.

/WINDOWS=n

Specifies the number of mapping pointers (used to access data in the file) to be allocated for file windows. The value can be an integer in the range of 7 through 80. The default is 7.

EXAMPLES

1
```
$ INITIALIZE/USER_NAME=CPA $FLOPPY1 ACCOUNTS
```

Initializes the volume on $FLOPPY1, labels the volume ACCOUNTS, and gives the volume a user name of CPA.

2
```
$ ALLOCATE DMA2:  TEMP
  _DMA2: ALLOCATED
$ INITIALIZE  TEMP:  BACK_UP_FILE
$ MOUNT  TEMP:  BACK_UP_FILE
%MOUNT-I-MOUNTED, BACK_UP_FILE mounted on _DMA2:
$ CREATE/DIRECTORY  TEMP:[ARCHIE]
```

The previous sequence of commands shows how to initialize an RK06/RK07 volume. First, the device is allocated, to ensure that no one else can access it. Then, when the volume is physically mounted on the device, the INITIALIZE command initializes it. When the volume is initialized, the MOUNT command makes the file structure available. Before you can place any files on the volume, you must create a directory, as shown by the CREATE/DIRECTORY command.

3
```
$ ALLOCATE MT:
  _MTB1:  ALLOCATED
$ INITIALIZE MTB1:  SOURCE
$ MOUNT MTB1:  SOURCE
%MOUNT-I-MOUNTED, SOURCE mounted on _MTB1:
$ COPY *.FOR  MTB1:
$ DIRECTORY MTB1:
   .
   .
   .
$ DISMOUNT MTB1:
```

These commands show the procedure necessary to initialize a magnetic tape. After allocating a drive, the magnetic tape is loaded on the device, and the INITIALIZE command writes the label SOURCE on it. Then, the MOUNT command mounts the magnetic tape so that files can be written on it.

INITIALIZE/QUEUE

Creates or initializes queues. You use this command to create queues and to assign them names and attributes.

Requires OPER privilege. Requires the /BATCH qualifier to create a batch queue.

FORMAT

INITIALIZE/QUEUE *queue-name[:]*

PARAMETER

queue-name[:]
Specifies the name of an execution queue or a generic queue. The queue name may be up to 31 alphanumeric characters.

DESCRIPTION

Initializing a Queue

Printer and batch queues are normally created by entering the necessary INITIALIZE command in a site-specific system start-up command procedure. However, once the system is running, you can use the INITIALIZE/QUEUE command to create additional queues as they are needed. The INITIALIZE /QUEUE command can also be used to update existing queue parameters of a stopped queue without affecting jobs in the queue.

To change a queue parameter for an existing queue, it is usually easier to use the SET QUEUE command. Use the INITIALIZE/QUEUE command to change queue parameters that are unavailable with SET QUEUE.

To initialize an existing queue, do the following: (1) stop the queue (use the STOP/QUEUE/NEXT command); (2) initialize the queue; and (3) start the queue. Once a queue has been stopped, you can specify new parameters to replace existing queue attributes. Unspecified parameters mean that these queue attributes remain as they were when the queue was previously initialized, started, or set.

To start the queue at the same time you initialize it, you can use the /START qualifier. Alternately, you can enter only the INITIALIZE/QUEUE command to get the queue ready and then later enter the START/QUEUE command to begin queue operations.

If the specified queue is already running, the INITIALIZE/QUEUE command is ignored. Use SET QUEUE to change the attributes of a running queue. Note that initializing an existing queue does not delete any jobs currently in that queue. Any new queue settings established by the new INITIALIZE /QUEUE command apply to all jobs waiting in the queue or subsequently entering the queue. Any jobs that are executing in the queue when it was stopped complete their execution under the old settings.

The following qualifiers apply to generic and execution queues:

/OWNER_UIC
/PROTECTION
/[NO]RETAIN
/[NO]START

These qualifiers apply to all types of execution queues:

/BASE_PRIORITY
/[NO]CHARACTERISTICS
/[NO]ENABLE_GENERIC
/ON
/WSDEFAULT
/WSEXTENT
/WSQUOTA

Qualifiers that apply only to batch execution queues are as follows:

/[NO]BATCH
/CPUDEFAULT
/CPUMAXIMUM
/[NO]DISABLE_SWAPPING
/JOB_LIMIT

Qualifiers that apply only to printer, terminal, or server queues are as follows:

/[NO]BLOCK_LIMIT
/[NO]DEFAULT
/FORM_MOUNTED
/[NO]LIBRARY
/[NO]PROCESSOR
/[NO]RECORD_BLOCKING
/SCHEDULE
/[NO]SEPARATE

The /[NO]GENERIC qualifier distinguishes a generic queue from an execution queue.

The /TERMINAL qualifier can be used only with generic terminal queues.

Types of Queues

There are several different types of queues on the system. In general, queues can be divided into two major types: generic and execution. When a job is sent to an execution queue, it is executed in that queue. No processing takes place in generic queues. Generic queues hold jobs that will execute on an execution queue when one is available.

There are four types of generic queues:

Generic batch queue	Holds batch jobs for processing on batch execution queues.
Generic printer queue	Holds print jobs for processing on printer execution queues.
Generic terminal queue	Holds print jobs for processing on terminal execution queues.
Generic server queue	Holds jobs for execution on server execution queues.

The system manager or operator uses the /GENERIC qualifier to specify which execution queues can be accessed by a generic queue. The /ENABLE_GENERIC qualifier can be used when initializing an execution queue to enable jobs to be placed in that queue by a generic queue even if that execution queue name was not specified with the /GENERIC qualifier of the generic queue.

There are several types of execution queues:

- Execution batch queue
- Output queue
  - –Printer queue
  - –Terminal queue
- Execution server queue

Execution batch queues execute batch jobs. Batch jobs are those that request the execution of one or more command procedures in a batch process.

Output queues execute print jobs. A print job requests the processing of one or more files by a symbiont executing in a symbiont process. The default system symbiont is designed to print files on hardcopy devices (printers or terminals). Customer-written symbionts can be designed for this or any other file processing activity.

Output queues include both printer and terminal execution queues. These execution queues execute print jobs: printer queues execute the jobs on print devices; terminal queues execute the jobs at terminals that have been designed for receiving print jobs.

Execution server queues execute jobs using the server processor specified with the /PROCESSOR qualifier. Server queue processors are customer-written.

Another type of queue is the logical queue. A logical queue is a special type of generic queue that can place work only into the execution queue specified in the ASSIGN/QUEUE command. The logical queue's relation to an execution queue remains in effect until a DEASSIGN/QUEUE command is issued which negates or changes the assignment.

QUALIFIERS

/BASE_PRIORITY=n

Specifies the base process priority at which jobs are initiated from a batch queue or the base priority of the symbiont process for a printer, terminal, or server queue. By default, if you omit the qualifier, jobs are initiated at the same priority as the base priority established by DEFPRI at system generation. The n specifier can be any decimal value from 0 to 15.

/BATCH
/NOBATCH (default)

Specifies that you are initializing a batch queue. If you are reinitializing an existing queue, you can use the /BATCH qualifier only if the queue was created as a batch queue.

A batch queue is classified as either an execution or generic queue. By default, the /BATCH qualifier initializes an execution queue. To specify a generic batch queue, use the /GENERIC qualifier together with the /BATCH qualifier.

The /BATCH and /DEVICE qualifiers are mutually exclusive; the /NOBATCH and /NODEVICE qualifiers also cannot be used together.

/BLOCK_LIMIT=([lowlim,]uplim) /NOBLOCK_LIMIT (default)

Limits the size of print jobs that can be executed on a printer or terminal queue. This qualifier allows you to reserve certain printers for certain size jobs. You must specify at least one of the parameters.

The lowlim parameter is a decimal number referring to the minimum number of blocks accepted by the queue for a print job. If a print job is submitted that contains fewer blocks than the lowlim value, the job remains pending until the block limit for the queue is changed, enabling it to execute.

The uplim parameter is a decimal number referring to the maximum number of blocks that will be accepted by the queue for a print job. If a print job is submitted that exceeds this value, the job remains pending until the block limit for the queue is changed, enabling it to execute.

If you specify only an upper limit for jobs, you can omit the parentheses. For example, /BLOCK_LIMIT=1000 means that only jobs with 1000 blocks or less execute in the queue. To specify only a lower job limit, you must use two double quotation marks to indicate the upper specifier. For example, /BLOCK_LIMIT=(500,"") means any job with 500 or more blocks executes in the queue. You can specify both a lower and upper limit. For example, /BLOCK_LIMIT=(200,2000) means that jobs with less than 200 blocks or more than 2000 blocks will not run in the queue.

The /NOBLOCK_LIMIT qualifier cancels the /BLOCK_LIMIT setting previously established for that queue.

/CHARACTERISTICS=(characteristic[,...]) /NOCHARACTERISTICS (default)

Specifies one or more characteristics for processing jobs on the queue. If only one characteristic is specified, you can omit the parentheses. Each time you specify /CHARACTERISTICS, all previously set characteristics are erased. Only the ones specified with the qualifier are now established for the queue.

Queue characteristics are installation-specific. The characteristic parameter can be either a value from 0 to 127 or a characteristic name that has been defined by the DEFINE/CHARACTERISTIC command.

When users include the /CHARACTERISTICS qualifier with a PRINT or SUBMIT command, all the characteristics they specify must also be specified for the queue that will be executing the job. If not, the job remains pending in the queue until the queue characteristics are changed, or until the entry is deleted with the DELETE/ENTRY command. Users need not specify every characteristic of a queue with a PRINT or SUBMIT command as long as the ones they specify are a subset of the characteristics set for that queue. The job also runs if no characteristics are specified.

The /NOCHARACTERISTICS qualifier cancels any /CHARACTERISTICS settings previously established for that queue.

/CLOSE

Prevents jobs from being entered in the queue through PRINT or SUBMIT commands or as a result of requeue operations. To allow jobs to be entered, use the /OPEN qualifier. Whether a queue accepts or rejects new job entries is independent of the queue's state (such as paused, stopped, stalled). When a

queue is marked closed, jobs executing continue to execute, and jobs already pending in the queue continue to be candidates for execution.

/CPUDEFAULT=time

Indicates the default CPU time limit for batch jobs. Time can be specified as a delta time, 0, NONE (the default), or INFINITE. You can specify up to 497 days of delta time. Both the value 0 and the keyword INFINITE allow unlimited CPU time (subject to the restrictions imposed by the /CPUMAXIMUM qualifier or the user authorization file).

/CPUMAXIMUM=time

Indicates the maximum CPU time limit for batch jobs. The /CPUMAXIMUM qualifier overrides the time limit specified in the user authorization file (UAF). Time can be specified as a delta time, 0, NONE (the default), or INFINITE. You can specify up to 497 days of delta time. Both the value 0 and the keyword INFINITE allow unlimited CPU time. Specify NONE when a maximum CPU time limit is not desired. Refer to Table DCL–1 for more information on specifying CPU time limits.

A CPU time limit for processes is specified by each user record in the system UAF. You can also specify the following: a default CPU time limit for all jobs in a given queue and a maximum CPU time limit for all jobs in a given queue. Table DCL–1 shows the action taken for each value specified and possible combinations of specifications.

Table DCL–1 CPU Time Limit Specifications and Actions

CPU Time Limit Specified By The SUBMIT Command?	Default CPU Time Limit Specified For The Queue?	Maximum CPU Time Limit Specified For The Queue?	Action Taken
No	No	No	Use the UAF value.
Yes	No	No	Use the smaller of SUBMIT command and UAF values
Yes	Yes	No	Use the smaller of SUBMIT command and UAF values
Yes	No	Yes	Use the smaller of SUBMIT command and queue's maximum values
Yes	Yes	Yes	Use the smaller of SUBMIT command and queue's maximum values

Table DCL–1 (Cont.) CPU Time Limit Specifications and Actions

CPU Time Limit Specified By The SUBMIT Command?	Default CPU Time Limit Specified For The Queue?	Maximum CPU Time Limit Specified For The Queue?	Action Taken
No	Yes	Yes	Use the smaller of queue's default and maximum values
No	No	Yes	Use the maximum value
No	Yes	No	Use the smaller of UAF and queue's default values

/DEFAULT=(option[,...])
/NODEFAULT

Establishes defaults for certain options of the PRINT command. Defaults are specified by the list of options. If you specify only one option, you can omit the parentheses. Once an option is set for the queue by the /DEFAULT qualifier, users do not have to specify that option in their PRINT commands. The /DEFAULT qualifier cannot be used with the /GENERIC qualifier. Possible options are as follows:

[NO]BURST[=keyword]	Specifies where to print burst pages (flag pages that are printed over the paper's perforations for easy identification of individual files in a print job). The keyword ALL places burst pages before each printed file in the job. The keyword ONE places a burst page before the first printed file in the job.
[NO]FEED	Specifies whether a form feed is automatically inserted at the end of a page. (The default is FEED.)
[NO]FLAG[=keyword]	Specifies where to print flag pages (containing the job entry number, the name of the user submitting the job, and so on). The keyword ALL places flag pages before each printed file in the job. The keyword ONE places a flag page before the first printed file in the job.
FORM=type	Specifies the default form for a printer, terminal, or server queue. If a job is not submitted with an explicit form definition, then this form will be used to process the job. The systemwide default form, form=0, is the default value for this keyword. See also /FORM_MOUNTED.

[NO]TRAILER[=keyword]	Specifies where to print trailer pages. The keyword ALL places trailer pages after each printed file in the job. The keyword ONE places a trailer page after the last printed file in the job.

If you specify any of the keywords BURST, FLAG, or TRAILER without specifying a value, the value ALL is used by default.

/DESCRIPTION=string
/NODESCRIPTION (default)

A string of up to 255 characters used to provide operator-supplied information about the queue.

If the string contains alphanumeric, underscore, or dollar sign characters it must be enclosed in quotation marks (").

The /NODESCRIPTION qualifier removes any descriptive text that may have been associated with the queue.

/DEVICE[=option]
/NODEVICE

Specifies that you are initializing an output queue of a particular type. If you are re-initializing an existing queue, you can use the /DEVICE qualifier only if the queue was created as an output queue. Possible options are as follows:

PRINTER	Indicates that this is a printer queue.
SERVER	Indicates that this is a server queue. An execution server queue is controlled by the user-modified or user-written symbiont specified with the /PROCESSOR qualifier.
TERMINAL	Indicates that this is a terminal queue.

The use of /DEVICE without designating a queue type is equivalent to specifying /DEVICE=PRINTER.

An output queue is classified as either an execution or generic queue. By default, the /DEVICE qualifier initializes an execution queue of the designated type. To specify a generic printer, server, or terminal queue, use the /GENERIC qualifier together with the /DEVICE qualifier.

For an output execution queue, the queue type you specify with the /DEVICE qualifier is for informational purposes. When the queue is started, the symbiont associated with the queue determines the actual queue type. The standard symbiont examines device characteristics to establish whether the queue should be marked as printer or terminal. By convention, user-modified and user-written symbionts mark the queue as a server queue.

The /DEVICE and /BATCH qualifiers are mutually exclusive; the /NODEVICE and /NOBATCH qualifiers also cannot be used together.

/DISABLE_SWAPPING
/NODISABLE_SWAPPING (default)

Controls whether batch jobs executed from a queue can be swapped in and out of memory.

/ENABLE_GENERIC (default)
/NOENABLE_GENERIC

Specifies whether files queued to a generic queue that does not have specific associated execution queues (named with the /GENERIC qualifier) can be placed in this execution queue for processing. (See the description of the /GENERIC qualifier for more information.)

/FORM_MOUNTED=type

Specifies the form type for a printer, terminal, or server queue. If the stock of the mounted form is not identical to the stock of the default form, as indicated by the DCL command qualifier /DEFAULT=FORM=type, all jobs submitted to this queue without an explicit form definition enter a pending state. If a job is submitted with an explicit form and the stock of the explicit form is not identical to the stock of the mounted form, the job enters a pending state. In both cases, the pending state is maintained until the stock of the mounted form of the queue is identical to the stock of the form associated with the job.

Specify the form type using either a numeric value or a form name that has been defined by the DEFINE/FORM command. Form types are installation-specific. The /FORM_MOUNTED qualifier is incompatible with the /GENERIC qualifier.

/GENERIC[=(queue-name[,...])]
/NOGENERIC (default)

Specifies that this is a generic queue and that jobs placed in it can be moved for processing to compatible execution queues. The /GENERIC qualifier optionally accepts a list of target execution queues that have been previously defined. For a generic batch queue, these target queues must be batch execution queues. For a generic output queue, these target queues must be output execution queues, but can be of any type (printer, server, or terminal). For example, a generic printer queue can feed a mixture of printer and terminal execution queues.

If you do not specify any target queues with the /GENERIC qualifier, jobs can be moved to any execution queue that (1) is initialized with the /ENABLE_GENERIC qualifier, and (2) is the same type (batch, printer, server, or terminal) as the generic queue. Moreover, for a generic server queue, an additional check is made: the symbiont named with the /PROCESSOR qualifier must be the same for both the generic and execution queues.

The /GENERIC qualifier is used in conjunction with either the /BATCH or /DEVICE qualifiers to define the queue as a generic batch, printer, server, or terminal queue. If neither /BATCH or /DEVICE is specified on creation of a generic queue, it becomes a generic printer queue by default.

/JOB_LIMIT=n

Indicates the number of batch jobs that can be executed concurrently from the queue. Specify a number in the range 0 to 255. The job limit default value for n is 1.

/LIBRARY=file-name
/NOLIBRARY

Specifies the file name for the device control library. When you are initializing an output queue, you can use the /LIBRARY qualifier to specify an alternate device control library. The default library is SYS$LIBRARY:SYSDEVCTL.TLB. Only a file name can be used as the

parameter of the /LIBRARY qualifier. The system always assumes that the location of the file is in SYS$LIBRARY and that the file type is TLB.

/ON=[node::]device[:] (printer, terminal, server queue)
/ON=node:: (batch queue)

Specifies the node or device, or both, on which this execution queue is located. For batch queues, only the node name can be specified. You can include both the node name and the device name for printer and terminal queues. By default, a queue executes on the same node from which you first start the queue. The default device parameter is the same as the queue name.

The node name is used only in VAXcluster systems; it must match the node name specified by the SYSGEN parameter SCSNODE for the processor on which the queue executes.

/OPEN (default)

Allows jobs to be entered in the queue through PRINT or SUBMIT commands or as the result of requeue operations. To prevent jobs from being entered, use the /CLOSE qualifier. Whether a queue accepts or rejects new job entries is independent of the queue's state (such as paused, stopped, stalled).

/OWNER_UIC=uic

Enables you to change the UIC of the queue. Specify the UIC using standard UIC format as described in Section 8.1 of the *VMS DCL Concepts Manual*. The default UIC is [1,4].

/PROCESSOR=file-name
/NOPROCESSOR

Allows users to specify their own print symbionts. The file name specifier can be any valid file name. Only a file name can be used as a parameter of the /PROCESSOR qualifier. The system supplies the device and directory name SYS$SYSTEM as well as the file type EXE.

If you use this qualifier for an output queue, it specifies that the symbiont image to be executed is SYS$SYSTEM:file-name.EXE. By default, SYS$SYSTEM:PRTSMB.EXE is executed. If you use this qualifier for a generic queue, it specifies that the generic queue can place jobs only on queues that have been established as server queues and that are executing the specified symbiont image.

The /NOPROCESSOR qualifier cancels the effect of a previous /PROCESSOR setting.

/PROTECTION=(codes)

Specifies the protection of the queue. Ownership categories are: SYSTEM, OWNER, GROUP, WORLD; each category can be abbreviated to its first character. Access categories are: R (READ), W (WRITE), E (EXECUTE), or D (DELETE); a null access specification means no access. The default protection is: (SYSTEM:E, OWNER:D, GROUP:R, WORLD:W). See Section 8.1 of the *VMS DCL Concepts Manual* for more information on specifying protection code. See the *Guide to Maintaining a VMS System* for more information on controlling queue operations through UIC-based protection.

/RECORD_BLOCKING (default)
/NORECORD_BLOCKING

Determines whether the symbiont can concatenate (or block together) output records for transmission to the output device. If you specify

/NORECORD_BLOCKING, the symbiont is directed to send each formatted record in a separate I/O request to the output device. For the standard VMS print symbiont, record blocking can have a significant performance advantage over single-record mode.

/RETAIN[=option]
/NORETAIN (default)

Holds jobs in the queue in a completed status after they have executed. The /NORETAIN qualifier enables you to reset the queue to the default. Possible options are as follows:

ALL (default)	Holds all jobs in the queue after execution
ERROR	Holds in the queue only jobs that complete unsuccessfully

/SCHEDULE=[NO]SIZE

Specifies whether pending jobs in a printer, terminal, or server queue are scheduled for printing based on the size of the job. When the default, /SCHEDULE=SIZE, is in effect, shorter jobs print before longer ones.

Note that if you enter this command while there are pending jobs in any queue, its effect on future jobs is unpredictable.

/SEPARATE=(option[,...])
/NOSEPARATE (default)

Specifies the job separation defaults for a printer or terminal queue. The /SEPARATE qualifier is incompatible with the /GENERIC qualifier. The job separation options are as follows:

[NO]BURST	Specifies whether a burst page prints at the beginning of every job. Specifying BURST also results in a flag page being printed.
[NO]FLAG	Specifies whether a flag page prints at the beginning of every job.
[NO]TRAILER	Specifies whether a trailer page prints at the end of every job.
[NO]RESET=(module[,...])	Specifies a job reset sequence for the queue. The specified modules from the device control library are used to reset the device each time a job reset occurs.

/START
/NOSTART (default)

Starts the queue being initialized by the current INITIALIZE/QUEUE command.

/TERMINAL
/NOTERMINAL (default)

Indicates that the output queue is a terminal queue. The /NOTERMINAL qualifier cancels the effect of a previous /TERMINAL qualifier on the same command. It is supported in this release for compatibility with VMS V4.n.

The function of the /[NO]TERMINAL qualifier has been superseded by the /[NO]DEVICE qualifier. DIGITAL recommends that you use this new qualifier and that existing command procedures using /[NO]TERMINAL be updated.

/WSDEFAULT=n

Defines a working set default for a batch job. The value set by this qualifier overrides the value defined in the user authorization file (UAF) of any user submitting a job to the queue.

Possible values are a positive integer in the range 1 through 65,535, 0, or the word NONE as the value for n. If 0 or NONE is specified for n, the working set default value defaults to the value specified either in the UAF or by the SUBMIT command (if specified). For more information, refer to Table DCL–2.

You can also specify this qualifier for an output queue. Used in this context, it establishes the working set default of the symbiont process for a printer, terminal, or server queue when the symbiont process is created.

/WSEXTENT=n

Defines a working set extent for the batch job. The value set by this qualifier overrides the value defined in the user authorization file (UAF) of any user submitting a job to the queue.

Possible values are a positive integer in the range 1 through 65,535, 0, or the word NONE as the value for n. If 0 or NONE is specified for n, the working set value defaults to the value specified either in the UAF or by the SUBMIT command (if specified). For more information, refer to Table DCL–2.

You can also specify this qualifier for an output queue. Used in this context, it establishes the working set extent of the symbiont process for a printer, terminal, or server queue when the symbiont process is created.

/WSQUOTA=n

Defines the working set page size (working set quota) for a batch job. The value set by this qualifier overrides the value defined in the user authorization file (UAF) of any user submitting a job to the queue.

Possible values are a positive integer in the range 1 through 65,535, 0, or the word NONE as the value for n. If 0 or NONE is specified for n, the working set quota defaults to the value specified either in the UAF or by the SUBMIT command (if specified). For more information, refer to Table DCL–2.

A working set default size and a working set quota (maximum size) are included in each user record in the system user authorization file (UAF), and can be specified for individual jobs or for all jobs in a given queue. The decision table (Table DCL–2) shows the action taken for different combinations of specifications that involve working set size and working set quota values.

You can also specify this qualifier for an output queue. Used in this context, it establishes the working set quota of the symbiont process for a printer, terminal, or server queue when the symbiont process is created.

Table DCL–2 Working Set Default, Extent, and Quota Decision

Value Specified By The SUBMIT COMMAND?	Value Specified For The Queue?	Action Taken
No	No	Use the UAF value
No	Yes	Use value for the queue
Yes	Yes	Use smaller of the two values
Yes	No	Compare specified value with UAF value; use the smaller

EXAMPLES

1
```
$ INITIALIZE/QUEUE/START/DEFAULT=FLAG SYS$PRINT/ON=LPA0:
$ INITIALIZE/QUEUE/START/BATCH/JOB_LIMIT=4 SYS$BATCH
```

In this example, the two commands initialize and start the printer queue SYS$PRINT on device LPA0 and then the batch queue SYS$BATCH. The /DEFAULT=FLAG qualifier causes a flag page to precede each file for jobs in the printer queue. The /JOB_LIMIT=4 qualifier allows as many as four batch jobs to be initiated concurrently from the batch queue. Both queues are started as soon as they have been initialized.

2
```
$ INITIALIZE/QUEUE/START/BATCH/JOB_LIMIT=3 SYS$BATCH
$ INITIALIZE/QUEUE/START/BATCH/JOB_LIMIT=1/WSEXTENT=2000 BIG_BATCH
$ INITIALIZE/QUEUE/START/DEFAULT=FORM=LN01_PORTRAIT LN01_PRINT
$ INITIALIZE/QUEUE/START/DEFAULT=(FLAG,TRAILER=ONE) LPA0:
$ INITIZLIZE/QUEUE/START/DEFAULT=(FLAG,TRAILER=ONE)/BLOCK_LIMIT=(1000,"") LPB0:
$ INITIALIZE/QUEUE/START/GENERIC=(LPA0,LPB0) SYS$PRINT
$ INITIALIZE/QUEUE/START/FORM_MOUNTED=LETTER/BLOCK_LIMIT=50 LQP /ON=TXA5:
```

In this example, the first INITIALIZE/QUEUE command creates a batch queue called SYS$BATCH that can be used for any batch job. The /JOB_LIMIT qualifier allows three jobs to execute concurrently. The second INITIALIZE/QUEUE command creates a second batch queue called BIG_BATCH that is designed for large jobs. Only one job can execute at a time. The working set extent can be as high as 2000.

The remaining INITIALIZE/QUEUE commands set up printer queues. The first creates the printer queue LN01_PRINT with the default form LN01_PORTRAIT. Both queue LPA0 and LPB0 are set to put flag and trailer pages between each file. If a job contains more than one file for printing, flag pages separate each file within the job. In addition, LPB0 has a minimum block size of 1000. Thus only print jobs larger than 1000 blocks can execute on that queue. SYS$PRINT is established as a generic queue that can direct jobs to either LPA0 or LPB0. Jobs that are too small to run on LPB0 will be queued from SYS$PRINT to LPA0.

The last INITIALIZE/QUEUE command sets up a terminal queue on TXA5. This queue is limited to PRINT commands that specify the form type LETTER. LETTER has been established at this site to indicate special letter-head paper. The block size limit is 50, indicating that this queue is reserved for jobs smaller than 51 blocks.

INQUIRE

Reads a value from SYS$COMMAND (usually the terminal in interactive mode or the next line in the main command procedure) and assigns it to a symbol.

FORMAT

INQUIRE *symbol-name [prompt-string]*

PARAMETERS

symbol-name

Specifies a 1- through 255-alphanumeric character symbol to be given a value.

prompt-string

Specifies the prompt to be displayed at the terminal when the INQUIRE command is executed. String values are automatically converted to uppercase. Also, any leading and trailing spaces and tabs are removed, and multiple spaces and tabs between characters are compressed to a single space.

Enclose the prompt in quotation marks (") if it contains lowercase characters, punctuation, multiple blanks or tabs, or an at sign (@). To denote an actual quotation mark in a prompt-string, enclose the entire string in quotation marks and use two consecutive quotation marks ("") within the string.

When the system displays the prompt string at the terminal, it generally places a colon (:) and a space at the end of the string. (See the /PUNCTUATION qualifier.)

If you do not specify a prompt string, the command interpreter uses the symbol name to prompt for a value.

DESCRIPTION

The INQUIRE command displays the prompting message to and reads the response from the input stream established when your process was created. This means that when the INQUIRE command is executed in a command procedure executed interactively, the prompting message is always displayed on the terminal, regardless of the level of nesting of command procedures. Note that input to the INQUIRE command in command procedures will be placed in the RECALL buffer.

When you enter a response to the prompt string, the value is assigned as a character string to the specified symbol. Lowercase characters are automatically converted to uppercase, leading and trailing spaces and tabs are removed, and multiple spaces and tabs between characters are compressed to a single space. To prohibit conversion to uppercase and retain space and tab characters, place quotation marks around the string.

To use symbols or lexical functions when you enter a response to the prompt string, use apostrophes to request symbol substitution.

Note that you can also use the READ command to obtain data interactively from the terminal. The READ command accepts data exactly as the user types it; characters are not automatically converted to uppercase and spaces are not compressed. However, symbols and lexical functions will not be translated even if you use apostrophes to request symbol substitution.

INQUIRE

When an INQUIRE command is entered in a batch job, the command reads the response from the next line in the command procedure; if procedures are nested, it reads the response from the first level command procedure. If the next line in the batch job command procedure begins with a dollar sign, the line is interpreted as a command, not as a response to the INQUIRE command. The INQUIRE command then assigns a null string to the specified symbol, and the batch job continues processing with the command on the line following the INQUIRE command.

QUALIFIERS

/GLOBAL

Specifies that the symbol be placed in the global symbol table. If you do not specify the /GLOBAL qualifier, the symbol is placed in the local symbol table.

/LOCAL (default)

Specifies that the symbol be placed in the local symbol table for the current command procedure.

/PUNCTUATION (default)
/NOPUNCTUATION

Inserts a colon (:) and a space after the prompt when it is displayed on the terminal. To suppress the colon and space, specify /NOPUNCTUATION.

EXAMPLES

1

```
$ INQUIRE CHECK "Enter Y[ES] to continue"
$ IF .NOT. CHECK THEN EXIT
```

The INQUIRE command displays the following prompting message at the terminal:

```
Enter Y[ES] to continue:
```

The INQUIRE command prompts for a value, which is assigned to the symbol CHECK. The IF command tests the value assigned to the symbol CHECK. If the value assigned to CHECK is true (that is, an odd numeric value, a character string that begins with a T, t, Y, or y, or an odd numeric character string), the procedure continues executing.

If the value assigned to CHECK is false (that is, an even numeric value, a character string that begins with any letter except T, t, Y, or y, or an even numeric character string), the procedure exits.

2

```
$ INQUIRE COUNT
$ IF COUNT .GT. 10 THEN GOTO SKIP
  .
  .
  .
$ SKIP:
```

The INQUIRE command prompts for a count with the message:

```
COUNT:
```

Then the command procedure uses the value of the symbol COUNT to determine whether to execute the next sequence of commands or to transfer control to the line labeled SKIP.

```
3  $ IF P1 .EQS. "" THEN INQUIRE P1 "FILE NAME"
   $ FORTRAN 'P1'
```

The IF command checks whether a parameter was passed to the command procedure by checking if the symbol P1 is null; if it is, it means that no parameter was specified, and the INQUIRE command is issued to prompt for the parameter. If P1 was specified, the INQUIRE command is not executed, and the FORTRAN command compiles the name of the file specified as a parameter.

INSTALL

Invokes the Install Utility, which enhances the performance of selected executable and shareable images by making them "known" to the system and assigning them appropriate attributes. For a complete description of the Install Utility, see the *VMS Install Utility Manual.*

FORMAT

INSTALL *[subcommand] [filespec]*

JOB

Identifies the beginning of a batch job submitted through a card reader. Each batch job submitted through the system card reader must be preceded by a JOB card.

JOB cannot be abbreviated.

FORMAT

$ JOB *user-name*

PARAMETER

user-name

Identifies the user name under which the job is to be run. Specify the user name as you would during the login procedure.

DESCRIPTION

The JOB card identifies the user submitting the job and is followed by a PASSWORD card giving the password. (Although the PASSWORD card is required, you do not have to use a password on the card if the account has a null password.)

The user name and password are validated by the system authorization file in the same manner as they are validated in the login procedure. The process that executes the batch job is assigned the disk and directory defaults and privileges associated with the user account. If a LOGIN.COM file exists for the specified user name, it is executed at the start of the job.

The end of a batch job is signaled by the EOJ command, by an EOF card (12-11-0-1-6-7-8-9 overpunch), or by another JOB card.

QUALIFIERS

/AFTER=time

Holds the job until the specified time. If the specified time has already passed, the job is queued for immediate processing.

The time can be specified as either an absolute time or a combination of absolute and delta times. See Section 1.4 of the *VMS DCL Concepts Manual* for complete information on specifying time values.

/CHARACTERISTICS=(characteristic[,...])

Specifies one or more characteristics required for processing the job. If you specify only one characteristic, you can omit the parentheses. Codes for characteristics are installation-defined. Use the SHOW QUEUE /CHARACTERISTICS command to see which characteristics are available on your system.

All the characteristics specified for the job must also be specified for the queue that will execute the job. If not, the job remains pending in the queue until the queue characteristics are changed or the entry is deleted with the DELETE/ENTRY command. Users need not specify every characteristic of a queue with the JOB command as long as the ones they specify are a subset of the characteristics set for that queue. The job also runs if no characteristics are specified.

/CLI=file-name

Specifies a different command language interpreter (CLI) with which to process the job. The file name specifies that the CLI be SYS$SYSTEM:filename.EXE. The default CLI is that defined in the user authorization file (UAF).

/CPUTIME=n

Specifies a CPU time limit for the batch job. Time can be specified as delta time, 0, NONE, or INFINITE. (See Section 1.4 of the *VMS DCL Concepts Manual* for information on specifying time values.)

When you need less CPU time than authorized, use the /CPUTIME qualifier to override the base queue value established by the system manager or the value authorized in your UAF. Specify 0 or INFINITE to request an infinite amount of time. Specify NONE when you want the CPU time to default to your UAF value or the limit specified on the queue. Note that you cannot request more time than permitted by the base queue limits or your UAF.

/DELETE (default)
/NODELETE

Controls whether the batch input file is deleted after the job is processed. If you specify /NODELETE, the file is saved in the user's default directory under the default name INPBATCH.COM. If you specify the /NAME qualifier, the file name of the batch input file is the same as the job name you supply with /NAME.

/HOLD
/NOHOLD (default)

Controls whether or not the job is to be made available for immediate processing.

If you specify /HOLD, the job is not released for processing until you specifically release it with the /NOHOLD or /RELEASE qualifier of the SET QUEUE/ENTRY command.

/KEEP
/NOKEEP (default)

Controls whether the log file is deleted after it is printed. /NOKEEP is the default unless /NOPRINTER is specified.

/LOG_FILE=file-spec
/NOLOG_FILE

Controls whether a log file with the specified name is created for the job or whether a log file is created.

When you use the /LOG_FILE qualifier, the system writes the log file to the file you specify. If you use /NOLOG_FILE, no log file is created. If you specify neither form of the qualifier, the log file is written to a file in your default directory that has the same file name as the first command file in the job and a file type of LOG. Using neither /LOG_FILE nor /NOLOG_FILE is the default.

You can use the /LOG_FILE qualifier to specify that the log file be written to a different device. Logical names that occur in the file specification are translated at the time the job is submitted. The process executing the batch job must have access to the device on which the log file will reside.

If you omit the /LOG_FILE qualifier and specify the /NAME qualifier, the log file is written to a file having the same file name as that specified by the /NAME qualifier and the file type LOG.

/NAME=job-name

Specifies a file name string to be used as the job name and as the file name for both the batch job log file and the command file. The job name must be 1 to 39 alphanumeric characters and must be a valid file name. The default log file name is INPBATCH.LOG; the default command file name is INPBATCH.COM.

/NOTIFY
/NONOTIFY (default)

Controls whether a message is broadcast to any terminal at which you are logged in, notifying you when your job completes or aborts.

/PARAMETERS=(parameter[,...])

Specifies from 1 through 8 optional parameters that can be passed to the command procedure. The parameters define values to be equated to the symbols P1 through P8 in the batch job. The symbols are local to the specified command procedure.

If you specify only one parameter, you can omit the parentheses.

The commas delimit individual parameters. If the parameter contains any spaces, special characters or delimiters, or lowercase characters, enclose it in quotation marks. Individual parameters cannot exceed 255 characters.

/PRINTER=queue-name
/NOPRINTER

Controls whether the job log file is queued to the specified queue for printing when the job is complete. The default print queue for the log file is SYS$PRINT.

If you specify /NOPRINTER, the /KEEP qualifier is assumed.

/PRIORITY=n

Requires OPER or alter priority (ALTPRI) privilege to raise the priority above the value of the SYSGEN parameter MAXQUEPRI.

Specifies the job scheduling priority for the specified job. The value of *n* is an integer from 0 through 255, where 0 is the lowest priority and 255 is the highest.

The default value for /PRIORITY is the value of the SYSGEN parameter DEFQUEPRI. No privilege is needed to set the priority lower than the MAXQUEPRI value.

The /PRIORITY qualifier has no effect on the process priority. The queue establishes the process priority.

/QUEUE=queue-name[:]

Specifies the name of the batch queue in which the job is to be entered. If you do not specify /QUEUE, the job is placed in the default system batch job queue, SYS$BATCH.

/RESTART
/NORESTART (default)

Specifies whether the job restarts after a system failure or a STOP/QUEUE /REQUEUE command.

/TRAILING_BLANKS (default)
/NOTRAILING_BLANKS

Controls whether input cards in the card deck are read in card image form or input records are truncated at the last nonblank character. By default, the system does not remove trailing blanks from records read through the card reader. Use the /NOTRAILING_BLANKS qualifier to request that input records be truncated.

/WSDEFAULT=n

Defines a working set default for the batch job; the /WSDEFAULT qualifier overrides the working set size specified in the user authorization file (UAF). *N* can be any positive integer from 1 to 65,535, 0, or the keyword NONE.

Use this qualifier to impose a value lower than the base queue value established by the system manager or lower than the value authorized in your UAF. A value of 0 or the keyword NONE sets the default value to the value specified either in your UAF or by the working set quota established for the queue. You cannot request a value higher than your default.

/WSEXTENT=n

Defines a working set extent for the batch job; the /WSEXTENT qualifier overrides the working set extent in the UAF. *N* can be any positive integer from 1 to 65,535, 0, or the keyword NONE.

To impose a lower value, use this qualifier to override the base queue value established by the system manager rather than the value authorized in your UAF. A value of 0 or the keyword NONE sets the default value either to the value specified in the UAF or working set extent established for the queue. You cannot request a value higher than your default.

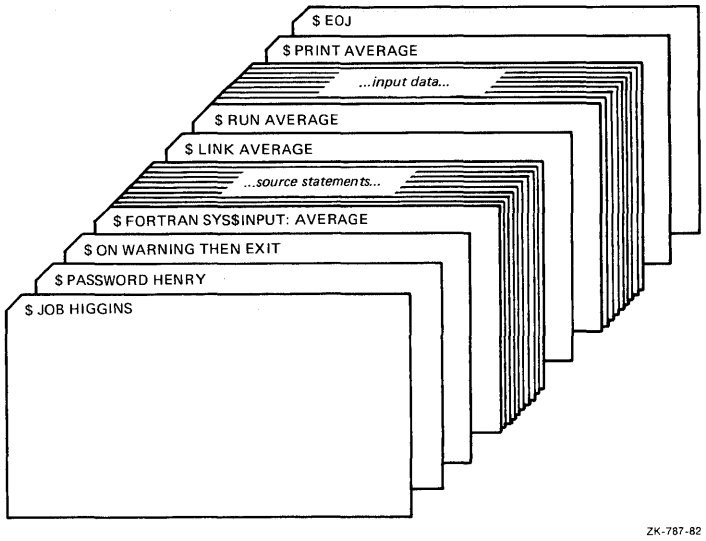
/WSQUOTA=n

Defines the maximum working set size (working set quota) for the batch job; the /WSQUOTA qualifier overrides the value in the UAF. *N* can be any positive integer from 1 to 65,535, 0, or the keyword NONE.

Use this qualifier to impose a value lower than the base queue value established by the system manager or lower than the value authorized in your user authorization file. Specify 0 or NONE if you want the working set quota defaulted to either your user authorization file value or the working set quota specified on the queue. You cannot request a value higher than your default.

EXAMPLES

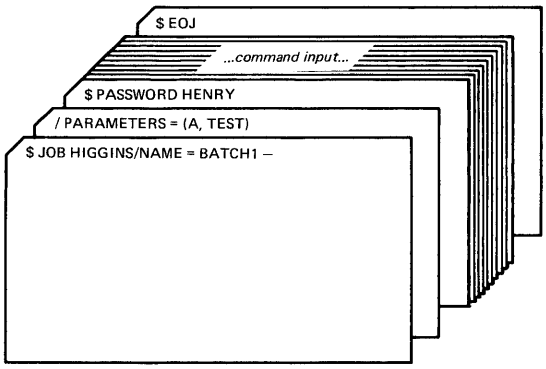
1



The JOB and PASSWORD cards identify and authorize the user HIGGINS to enter batch jobs. The command stream consists of a FORTRAN command and FORTRAN source statements to be compiled. The file name AVERAGE following the device name SYS$INPUT provides the compiler with a file name for the object and listing files. The output files are cataloged in user HIGGINS's default directory.

If the compilation is successful, the LINK command creates an executable image and the RUN command executes it. Input for the program follows the RUN command in the command stream. The last command in the job prints the program listing. The last card in the deck contains the EOJ (end of job) command.

2



ZK-788-82

The /NAME qualifier on the JOB card specifies a name for the batch job. When the job completes, the printed log file is identified as BATCH1.LOG. The JOB command is continued onto a second card with the continuation character (-). The /PARAMETERS qualifier defines P1 as A and P2 as TEST. The last card in the deck contains the EOJ (end of job) command.

Lexical Functions

A set of functions that return information about character strings and attributes of the current process.

DESCRIPTION

The command language includes constructs, called lexical functions, that return information about the current process and about arithmetic and string expressions. The functions are called lexical functions because the command interpreter evaluates them during the command input scanning (or lexical processing) phase of command processing.

You can use lexical functions in any context in which you normally use symbols or expressions. In command procedures, you can use lexical functions to translate logical names, perform character string manipulations, and determine the current processing mode of the procedure.

The general format of a lexical function is as follows:

F$function-name([args,...])

F$	Indicates that what follows is a lexical function.
function-name	A keyword specifying the function to be evaluated. Function names can be truncated to any unique abbreviation.
()	Enclose function arguments, if any. The parentheses are required for all functions, including functions that do not accept any arguments.
args,...	Specify arguments for the function, if any, using integer or character string expressions.

See Chapter 5 of the *VMS DCL Concepts Manual* for more information on specifying expressions.

Table DCL–3 lists each lexical function and briefly describes the information that each function returns. A detailed description of each function, including examples, is given in the following pages.

Table DCL–3 Summary of Lexical Functions

Function	Description
F$CVSI	Extracts bit fields from character string data and converts the result, as a signed value, to an integer.
F$CVTIME	Retrieves information about an absolute, combination, or delta time string.
F$CVUI	Extracts bit fields from character string data and converts the result, as an unsigned value, to an integer.
F$DIRECTORY	Returns the current default directory name string.
F$EDIT	Edits a character string based on the edits specified.

Table DCL–3 (Cont.) Summary of Lexical Functions

Function	Description
F$ELEMENT	Extracts an element from a string in which the elements are separated by a specified delimiter.
F$ENVIRONMENT	Obtains information about the DCL command environment.
F$EXTRACT	Extracts a substring from a character string expression.
F$FAO	Invokes the $FAO system service to convert the specified control string to a formatted ASCII output string.
F$FILE_ATTRIBUTES	Returns attribute information for a specified file.
F$GETDVI	Invokes the $GETDVI system service to return a specified item of information for a specified device.
F$GETJPI	Invokes the $GETJPI system service to return accounting, status, and identification information for a process.
F$GETQUI	Invokes the $GETQUI system service to return information about queues, batch and print jobs currently in those queues, form definitions, and characteristic definitions kept in the system job queue file.
F$GETSYI	Invokes the $GETSYI system service to return status and identification information about the local system, or about a node in the local cluster, if your system is part of a cluster.
F$IDENTIFIER	Converts an identifier in named format to its integer equivalent, or vice versa.
F$INTEGER	Returns the integer equivalent of the result of the specified expression.
F$LENGTH	Returns the length of a specified string.
F$LOCATE	Locates a character or character substring within a string and returns its offset within the string.
F$LOGICAL	Translates a logical name and returns the equivalence name string. (Superseded in function by F$TRNLNM.)
F$MESSAGE	Returns the message text associated with a specified system status code value.
F$MODE	Shows the mode in which a process is executing.
F$PARSE	Invokes the $PARSE RMS service to parse a file specification and return either the expanded file specification or the particular file specification field that you request.
F$PID	For each invocation, returns the next process identification number in sequence.

Table DCL–3 (Cont.) Summary of Lexical Functions

Function	Description
F$PRIVILEGE	Returns a value of "TRUE" or "FALSE" depending on whether your current process privileges match the privileges listed in the argument.
F$PROCESS	Returns the current process name string.
F$SEARCH	Invokes the $SEARCH RMS service to search a directory file, and returns the full file specification for a file you name.
F$SETPRV	Sets the specified privileges and returns a list of keywords indicating the previous state of these privileges for the current process.
F$STRING	Returns the string equivalent of the result of the specified expression.
F$TIME	Returns the current date and time of day, in the format dd-mm-yyy hh:mm:ss.cc.
F$TRNLNM	Translates a logical name and returns the equivalence name string or the requested attributes of the logical name.
F$TYPE	Determines the data type of a symbol.
F$USER	Returns the current user identification code (UIC).
F$VERIFY	Returns the integer 1 if command procedure verification is set on; returns the integer 0 if command procedure verification is set off. The F$VERIFY function also can set new verification states.

F$CVSI

Converts the specified bits in the specified character string to a signed number.

FORMAT

F$CVSI*(start-bit,number-of-bits,string)*

return value

The integer equivalent of the extracted bit field, converted as a signed value.

ARGUMENTS

start-bit

The offset of the first bit to be extracted. The low-order (rightmost) bit of a string is position number 0 for determining the offset. Specify the offset as an integer expression.

If you specify an expression with a negative value, or with a value that exceeds the number of bits in the string, then DCL displays the INVRANGE error message.

number-of-bits

The length of the bit string to be extracted, which must be less than or equal to the number of bits in the string.

If you specify an expression with a negative value, or with a value that is invalid when added to the bit position offset, then DCL displays the INVRANGE error message.

string

The string from which the bits are taken. Specify the string as a character string expression.

EXAMPLES

1

```
$ A[0,32] = %X2B
$ SHOW SYMBOL A
  A = "+..."
$ X = F$CVSI(0,4,A)
$ SHOW SYMBOL X
  X = -5   Hex = FFFFFFFB  Octal = 37777777773
```

This example uses an arithmetic overlay to assign the hexadecimal value 2B to all 32 bits of the symbol A. See the description of the Assignment Statement for more information on arithmetic overlays.

The symbol A has a string value after the overlay because it was previously undefined. (If a symbol is undefined, it has a string value as a result of an arithmetic overlay. If a symbol was previously defined, it retains the same data type after the overlay.) The hexadecimal value 2B corresponds to the ASCII value of the plus sign (+).

Next, the F$CVSI function extracts the low-order 4 bits from the symbol A; the low order 4 bits contain the binary representation of the hexadecimal value B. These bits are converted, as a signed value, to an integer. The converted value, –5, is assigned to the symbol X.

2
```
$ SYM[0,32] = %X2A
$ SHOW SYMBOL SYM
  SYM = "*..."
$ Y = F$CVSI(0,33,SYM)
%DCL-W-INVRANGE, field specification is out of bounds - check sign and size
$ SHOW SYMBOL Y
%DCL-W-UNDSYM, undefined symbol - check spelling
```

In this example, the width argument specified with the F$CVSI function is too large. Therefore, DCL issues an error message and the symbol Y is not assigned a value.

F$CVTIME

Converts an absolute or a combination time string to a string of the form "yyyy-mm-dd hh:mm:ss.cc". The F$CVTIME function can also return information about an absolute, combination, or delta time string.

FORMAT

F$CVTIME*([input_time] [,output_time_format] [,output_field])*

return value

A character string containing the requested information.

ARGUMENTS

input_time

Specifies a string containing an absolute, combination, or delta time, or TODAY, TOMORROW, or YESTERDAY. Specify the input time string as a character string expression. If the input_time argument is omitted or specified as a null string (""), the current system date and time, in absolute format, is used. If parts of the date field are omitted, the missing values default to the current date. If parts of the time field are omitted, the missing values default to zero.

See Section 1.4 of the *VMS DCL Concepts Manual* for more information on these time formats.

If the input_time argument is a delta time, you must specify the output_time_format argument as DELTA.

output_time_format

Specifies the time format for the information you want returned. Specify the output_time_format argument as one of the following character string expressions:

ABSOLUTE	The requested information should be returned in absolute time format, which is "dd-mmm-yyyy hh:mm:ss.cc".
COMPARISON (default)	The requested information should be returned in the form "yyyy-mm-dd hh:mm:ss.cc"; used for comparing two times.
DELTA	The requested information should be returned in delta format, which is "dddd-hh:mm:ss.cc". If the input_time argument is a delta time, the output_time_format argument must be DELTA.

output_field

Specifies a character string expression containing one of the following (do not abbreviate): DATE, MONTH, DATETIME (default), SECOND, DAY, TIME, HOUR, WEEKDAY, HUNDREDTH, YEAR, MINUTE. The information is returned in the time format specified by the output_time_format argument. If the input_time argument is a delta time and the output_time_format argument is DELTA, you cannot specify MONTH, WEEKDAY, or YEAR.

When the weekday is returned, the first letter is in uppercase, and the following letters are in lowercase.

DESCRIPTION

When using the F$CVTIME function, you can omit optional arguments that can be used to the right of the last argument you specify. However, you must include commas as placeholders if you omit optional arguments to the left of the last argument you specify.

When specifying the input time argument in either absolute or combination time format, you can specify ABSOLUTE or COMPARISON as the output time format argument; you cannot specify DELTA.

When specifying the input time argument in delta time format, you must specify DELTA as the output time format argument.

EXAMPLES

1

```
$ TIME = F$TIME()
$ SHOW SYMBOL TIME
  TIME = "15-APR-1988 10:56:23.10"
$ TIME = F$CVTIME(TIME)
$ SHOW SYMBOL TIME
  TIME = "1988-04-15 10:56:23.10"
```

This example uses the F$TIME function to return the system time as a character string and to assign the time to the symbol TIME. Then the F$CVTIME function is used to convert the system time to an alternate time format. Note that you do not need to place quotation marks around the argument TIME because it is a symbol. Symbols are automatically evaluated when they are used as arguments for lexical functions.

You can use the resultant string to compare two dates (using .LTS. and .GTS. operators). For example, you can use F$CVTIME to convert two time strings and store the results in the symbols TIME_1 and TIME_2. You can compare the two values, and branch to a label, based on the following results:

```
$ IF TIME_1 .LTS. TIME_2 THEN GOTO FIRST
```

2

```
$ NEXT = F$CVTIME("TOMORROW",,"WEEKDAY")
$ SHOW SYMBOL NEXT
 NEXT = "Tuesday"
```

In this example, the F$CVTIME returns the weekday that corresponds to the absolute time keyword "TOMORROW". You must enclose the arguments "TOMORROW" and "WEEKDAY" in quotation marks because they are character string expressions. Also, you must include a comma as a placeholder for the output time argument that is omitted.

F$CVUI

Extracts bit fields from character string data and converts the result to an unsigned number.

FORMAT

F$CVUI*(start-bit,number-of-bits,string)*

return value

The integer equivalent of the extracted bit field, converted as an unsigned value.

ARGUMENTS

start-bit

Specifies the offset of the first bit to be extracted. The low-order (rightmost) bit of a string is position number 0 for determining the offset. Specify the offset as an integer expression.

If you specify an expression with a negative value, or with a value that exceeds the number of bits in the string, DCL displays the INVRANGE error message.

number-of-bits

Specifies the length of the bit-string to be extracted, which must be less than or equal to the number of bits in the string argument.

If you specify an expression with a negative value, or with a value that is invalid when added to the bit position offset, DCL displays the INVRANGE error message.

string

Specifies the character string to be edited.

EXAMPLE

```
$ A[0,32] = %X2B
$ SHOW SYMBOL A
  A = "+ . . .  "
$ X = F$CVUI(0,4,A)
$ SHOW SYMBOL X
  X = 11   Hex = 0000000B  Octal = 00000000013
```

This example uses an arithmetic overlay to assign the hexadecimal value 2B to all 32 bits of the symbol A. The symbol A has a string value after the overlay because it was previously undefined. (If a symbol is undefined, it has a string value as a result of an arithmetic overlay. If a symbol was previously defined, it retains the same data type after the overlay.) The hexadecimal value 2B corresponds to the ASCII character "+".

Next, the F$CVUI function extracts the low-order 4 bits from the symbol A; the low-order 4 bits contain the binary representation of the hexadecimal value B. These bits are converted, as a signed value, to an integer. The converted value, 11, is assigned to the symbol X.

F$DIRECTORY

Returns the current default directory name string. The F$DIRECTORY function has no arguments, but must be followed by parentheses.

FORMAT

F$DIRECTORY()

return value

A character string for the current default directory name, including square brackets ([]). If you use the SET DEFAULT command and specify angle brackets (<>) in a directory specification, the F$DIRECTORY function returns angle brackets in the directory string.

ARGUMENTS

None.

DESCRIPTION

You can use the F$DIRECTORY function to save the name of the current default directory in a command procedure, change the default to another directory to do work, and later restore the original setting.

EXAMPLE

```
$ SAVE_DIR = F$DIRECTORY()
$ SET DEFAULT [MALCOLM.TESTFILES]
    .
    .
    .
$ SET DEFAULT 'SAVE_DIR'
```

This example shows an excerpt from a command procedure that uses the F$DIRECTORY function to save the current default directory setting. The assignment statement equates the symbol SAVE_DIR to the current directory. Then the SET DEFAULT command establishes a new default directory. Later, the symbol SAVE_DIR is used in the SET DEFAULT command that restores the original default directory.

Note that you can use the F$ENVIRONMENT function with the DEFAULT keyword to return the default disk and directory. You should use the F$ENVIRONMENT function rather than the F$DIRECTORY function in situations involving more than one disk.

F$EDIT

Edits the character string based on the edits specified in the edit-list.

FORMAT

F$EDIT*(string, edit-list)*

return value

A character string containing the specified edits.

ARGUMENTS

string

A character string to be edited. Quoted sections of the string are not edited.

edit-list

A character string containing one or more of the following keywords that specify the types of edits to be made to the string. If you use a list of keywords, separate them with commas. Do not abbreviate these keywords.

Edit	Action
COLLAPSE	Removes all spaces or tabs
COMPRESS	Replaces multiple spaces or tabs with a single space
LOWERCASE	Changes all uppercase characters to lowercase
TRIM	Removes leading and trailing spaces or tabs
UNCOMMENT	Removes comments
UPCASE	Changes all lowercase characters to uppercase

Edits are not applied to quoted sections of strings. Therefore, if a string contains quotation marks, the characters within the quotation marks are not affected by the edits specified in the edit list.

EXAMPLES

1

```
$ LINE = "    THIS    LINE    CONTAINS A ""  QUOTED  "" WORD"
$ SHOW SYMBOL LINE
  LINE =  "    THIS    LINE    CONTAINS A "  QUOTED  " WORD"
$ NEW_LINE = F$EDIT(LINE, "COMPRESS, TRIM")
$ SHOW SYMBOL NEW_LINE
  NEW_LINE = "THIS LINE CONTAINS A "  QUOTED  " WORD"
```

This example uses the F$EDIT function to compress and trim a string by replacing multiple blanks with a single blank, and by removing leading and trailing blanks. The string LINE contains quotation marks around the word QUOTED. (To enter quotation marks into a character string, use double quotations in the assignment statement.)

Note that the F$EDIT function does not compress the spaces in the quoted section of the string; therefore, the spaces are retained around the word QUOTED.

```
2   $ LOOP:
    $       READ/END_OF_FILE = DONE INPUT_FILE RECORD
    $       RECORD = F$EDIT(RECORD, "TRIM, UPCASE")
    $       WRITE OUTPUT_FILE RECORD
    $       GOTO LOOP
        .
        .
        .
```

This example sets up a loop to read records from a file, edit them, and write them to an output file. The edited records have leading and trailing blanks removed, and are converted to uppercase.

F$ELEMENT

Extracts one element from a string of elements.

FORMAT

F$ELEMENT*(element-number, delimiter, string)*

return value

A character string containing the specified element.

ARGUMENTS

element-number

The number of the element to extract (numbering begins with zero). Specify the element-number argument as an integer expression. If the element-number argument exceeds the number of elements in the string, F$ELEMENT returns the delimiter.

delimiter

A character used to separate the elements in the string. Specify the delimiter as a character string expression.

string

A string containing a delimited list of elements. Specify the string as a character string expression

EXAMPLES

1

```
$ DAY_LIST = "MON/TUE/WED/THU/FRI/SAT/SUN"
$ INQUIRE DAY "ENTER DAY (MON TUE WED THU FRI SAT SUN)"
$ NUM = 0
$ LOOP:
$       LABEL = F$ELEMENT(NUM,"/",DAY_LIST)
$       IF LABEL .EQS. "/" THEN GOTO ERROR
$       IF DAY .EQS. LABEL THEN GOTO 'LABEL'
$       NUM = NUM +1
$       GOTO LOOP
$
$ MON:
    .
    .
    .
```

This example sets up a loop to test an input value against the elements in a list of values. If the value for DAY matches one of the elements in DAY_LIST, control is passed to the corresponding label. If the value returned by the F$ELEMENT function matches the delimiter, the value DAY was not present in the DAY_LIST, and control is passed to the label ERROR.

2

```
$ ! INDEX.COM
$ !
$ CHAPTERS = "0,1,2,3,4,5,6,A,B,C"
$ NEXT = 0
$ LOOP:
$   NEXT = NEXT + 1
$   NUM = F$ELEMENT(NEXT,",",CHAPTERS)
$   RUN INDEX CHAP'NUM'
$   IF (CHAPTERS .NES. ",") THEN GOTO LOOP
```

This example processes files named CHAP1, CHAP2, ... CHAP6, CHAPA, CHAPB, and CHAPC, in that order. (0 is included in the CHAPTERS string to clarify the procedure logic.)

F$ENVIRONMENT

Returns information about the current DCL command environment.

FORMAT

F$ENVIRONMENT*(item)*

return value

Information that corresponds to the specified item. The return value can be either an integer or a character string, depending on the specified item.

ARGUMENT

item

A keyword, specified as a character string, that specifies the type of information to be returned. Do not abbreviate these keywords. Specify one of the following keywords:

Item	Data Type	Information Returned
CAPTIVE	string	TRUE if you are logged into a captive account. The system manager can define captive accounts in the user authorization file with the Authorize Utility.
CONTROL	string	Control characters currently enabled with SET CONTROL. Multiple characters are separated by commas; if no control characters are enabled, the null string ("") is returned.
DEFAULT	string	Current default device and directory name. The returned string is the same as SHOW DEFAULT output.
DEPTH	integer	Current command procedure depth. The command procedure depth is 0 when you log in interactively and when you submit a batch job. The command procedure depth is 1 when you execute a command procedure interactively or from within a batch job. A nested command procedure has a depth of 1 greater than the depth of the command procedure from which the nested procedure is executed.
INTERACTIVE	string	TRUE if the process is executing interactively.
KEY_STATE	string	Current locked keypad state. See the description of the DEFINE/KEY command for more information on keypad states.
MAX_DEPTH	integer	Maximum allowable command procedure depth.

Item	Data Type	Information Returned
MESSAGE	string	Current setting of SET MESSAGE qualifiers. Each qualifier in the string is prefaced by a slash; therefore, the output from F$ENVIRONMENT("MESSAGE") can be appended to the SET MESSAGE command to form a valid DCL command line.
NOCONTROL	string	Control characters currently disabled with SET NOCONTROL. Multiple characters are separated by commas; if no control characters are disabled, the null string is returned.
ON_CONTROL_Y	string	If issued from a command procedure, returns TRUE if ON_CONTROL_Y is set. ON_CONTROL_Y always returns FALSE at DCL command level.
ON_SEVERITY	string	If issued from a command procedure, returns the severity level at which the action specified with the ON command is performed. ON_SEVERITY returns "NONE" when SET NOON is in effect or at DCL command level.
OUTPUT_RATE	string	Delta time string containing the default output rate, which indicates how often data is written to the batch job log file while the batch job is executing. OUTPUT_RATE returns a null string if used interactively.
PROCEDURE	string	File specification of the current command procedure. PROCEDURE returns a null string if used interactively.
PROMPT	string	Current DCL prompt.
PROMPT_CONTROL	string	TRUE if a carriage return and line feed precede the prompt.
PROTECTION	string	Current default file protection.
SYMBOL_SCOPE	string	[NO]LOCAL,[NO]GLOBAL to indicate the current symbol scoping state. The string is in a form that can be used with the SET PROTECTION/DEFAULT command to form a valid DCL command line.
VERIFY_IMAGE	string	TRUE if image verification (SET VERIFY=IMAGE) is in effect. If image verification is in effect, then the command procedure echoes input data read by images.
VERIFY_PROCEDURE	string	TRUE if procedure verification SET VERIFY=PROCEDURE is in effect. If command verification is in effect, then the command procedure echoes DCL command lines.

EXAMPLES

1
```
$ SAVE_MESSAGE = F$ENVIRONMENT("MESSAGE")
$ SET MESSAGE/NOFACILITY/NOIDENTIFICATION
   .
   .
   .
$ SET MESSAGE'SAVE_MESSAGE'
```

This example uses the F$ENVIRONMENT function to save the current message setting before changing the setting. At the end of the command procedure, the original message setting is restored. The apostrophes surrounding the symbol SAVE_MESSAGE indicate that the value for the symbol should be substituted.

2
```
$ MAX = F$ENVIRONMENT("MAX_DEPTH")
$ SHOW SYMBOL MAX
  MAX = 32   Hex = 00000020  Octal = 00000000040
```

This example uses the F$ENVIRONMENT function to determine the maximum depth allowable within command procedures.

3
```
$ SAVE_PROT = F$ENVIRONMENT("PROTECTION")
$ SET PROTECTION = (SYSTEM:RWED, OWNER:RWED, GROUP, WORLD)/DEFAULT
   .
   .
   .
$ SET PROTECTION = ('SAVE_PROT')/DEFAULT
```

This example uses the F$ENVIRONMENT function to save the current default protection before changing the protection. At the end of the command procedure, the original protection is restored. You must place apostrophes around the symbol SAVE_PROT to request symbol substitution.

F$EXTRACT

Extracts the specified characters from the specified string.

FORMAT

F$EXTRACT*(start,length,string)*

return value

A character string containing the characters delimited by the start and length arguments.

ARGUMENTS

start

Specifies the offset of the starting character of the string you want to extract. Specify the start argument as an integer expression that is greater than or equal to 0.

The offset is the relative position of a character or a substring with respect to the beginning of the string. Offset positions begin with 0. The string always begins with the leftmost character.

If you specify an offset that is greater than or equal to the length of the string, F$EXTRACT returns a null string ("").

length

Specifies the number of characters you want to extract; must be less than or equal to the size of the string. Specify the length as an integer expression that is greater than or equal to 0.

If you specify a length that exceeds the number of characters from the offset to the end of the string, the F$EXTRACT returns the characters from the offset through the end of the string.

string

Specifies the character string to be edited. Specify the string as a character string expression.

EXAMPLES

1

```
$ NAME = "JOE SMITH"
$ FIRST = F$EXTRACT(0,3,NAME)
$ SHOW SYMBOL FIRST
  FIRST = "JOE"
```

This portion of a command procedure uses the F$EXTRACT function to extract the first three characters from the character string assigned to the symbol NAME. The offset and length arguments are integers, and the string argument is a symbol. You do not need to use quotations around integers or symbols when they are used as arguments for lexical functions.

2

```
$ P1 = "MYFILE.DAT"
$ FILENAME = F$EXTRACT(0,F$LOCATE(".",P1),P1)
```

This portion of a command procedure shows how to locate a character within a string, and how to extract a substring ending at that location.

The lexical function F$LOCATE gives the numeric value representing the offset position of a period in the character string value of P1. (The offset position of the period is equal to the length of the substring before the period.)

This F$LOCATE function is used as an argument in the F$EXTRACT function to specify the number of characters to extract from the string. If a procedure is invoked with the parameter MYFILE.DAT, these statements result in the symbol FILENAME being given the value MYFILE.

Note that the F$LOCATE function in the above example assumes that the file specification does not contain a node name or a directory specification containing a subdirectory name. To obtain the file name from a full file specification, use the F$PARSE function.

3

```
$ IF F$EXTRACT(12,2,F$TIME()) .GES. "12" THEN GOTO AFTERNOON
$ MORNING:
$ WRITE SYS$OUTPUT "Good morning!"
$ EXIT
$ AFTERNOON:
$ WRITE SYS$OUTPUT "Good afternoon!"
$ EXIT
```

This example shows a procedure that displays a different message, depending on whether the current time is morning or afternoon. It first obtains the current time of day by using the F$TIME function. The F$TIME function returns a character string, which is the string argument for the F$EXTRACT function. The F$TIME function is automatically evaluated when it is used as an argument, so you do not need to use quotation marks.

Next, the F$EXTRACT function extracts the hours from the date and time string returned by F$TIME. The string returned by F$TIME always contains the hours field beginning at an offset of 12 characters from the start of the string.

The F$EXTRACT function extracts two characters from the string, beginning at this offset, and compares the string value extracted with the string value 12. If the comparison is true, then the procedure writes "Good afternoon!". Otherwise, it writes "Good morning!".

Note that you can also use the F$CVTIME function to extract the hour field from a time specification. This method is easier than the one shown in the above example.

F$FAO

Invokes the $FAO system service to convert character and numeric input to character strings. (FAO stands for formatted ASCII output.) By specifying formatting instructions, you can use the F$FAO function to convert integer values to character strings, insert carriage returns and form feeds, insert text, and so on.

The F$FAO function converts the integer values into ASCII representations of their decimal, hexadecimal, and octal equivalents and substitutes the results into the output string.

FORMAT

F$FAO*(control-string[,arg1,arg2...arg15])*

return value

A character string containing formatted ASCII output. This output string is created from the fixed text and FAO directives in the control string.

ARGUMENTS

control-string

Specifies the fixed text of the output string, consisting of text and any number of FAO directives. The control string may be any length. Specify the control string as a character string expression.

The F$FAO function uses FAO directives to modify or insert ASCII data into the fixed text in the control string.

Table DCL–4 lists the FAO directives you can specify in a control string.

arg1,arg2...arg15

Specifies the arguments required by the FAO directives used in the control string. Specify the arguments arg1,arg2...arg15 as integer or character string expressions. Table DCL–4 lists the argument types required by each FAO directive.

If you specify an argument whose type (integer or string) does not match that of the corresponding directive, unpredictable results are returned. You can use the F$INTEGER and F$STRING lexical functions to convert arguments to the proper type.

FAO directives may require one or more arguments. The order of the arguments must correspond exactly with the order of the directives in the control string. In most cases, an error message is not displayed if you misplace an argument.

If there are not enough arguments listed, F$FAO continues reading past the end of an argument list. Therefore, always be sure to include enough arguments to satisfy the requirements of all the directives in a control string.

DESCRIPTION

Specify an FAO directive using any one of the following formats:

Format	Function
!DD	One directive
!n(DD)	A directive repeated a specified number of times
!lengthDD	A directive that places its output in a field of a specified length
!n(lengthDD)	A directive that is repeated a specified number of times and generates output fields of a specified length

The exclamation point (!) indicates that the following character or characters are to be interpreted as an FAO directive. DD represents a one- or two-character uppercase code indicating the action that F$FAO is to perform. When specifying repeat counts, n is a decimal value specifying the number of times the directive is to be repeated. The length value is a decimal value that instructs F$FAO to generate an output field of "length" characters.

Repeat counts and output lengths may also be specified by using a number sign (#) in place of an absolute numeric value. If you use a number sign (#), you must specify the numeric value as an integer expression in the corresponding place in the argument list.

When a variable output field is specified with a repeat count, only one length parameter is required, because each output string has the specified length.

The FAO directives are grouped in the following categories:

- Character string insertion
- Zero-filled numeric conversion
- Blank-filled numeric conversion
- Special formatting
- Parameter interpretation

Table DCL–4 summarizes the FAO directives and shows the required argument types. In addition, the following sections describe output strings from directives that perform character string insertion, zero-filled numeric conversion, and blank-filled numeric conversion.

Table DCL–4 Summary of FAO Directives

Directive	Argument Type	Description
Character string insertion:		
!AS	String	Inserts a character string as is
Zero-filled numeric conversion:		
!OB	Integer	Converts a byte to octal notation
!OW	Integer	Converts a word to octal notation
!OL	Integer	Converts a longword to octal notation
!XB	Integer	Converts a byte to hexadecimal notation

Table DCL–4 (Cont.) Summary of FAO Directives

Directive	Argument Type	Description
!XW	Integer	Converts a word to hexadecimal notation
!XL	Integer	Converts a longword to hexadecimal notation
!ZB	Integer	Converts a byte to decimal notation
!ZW	Integer	Converts a word to decimal notation
!ZL	Integer	Converts a longword to decimal notation
Blank-filled numeric conversion:		
!UB	Integer	Converts a byte to decimal notation without adjusting for negative numbers
!UW	Integer	Converts a word to decimal notation without adjusting for negative numbers
!UL	Integer	Converts a longword to decimal notation without adjusting for negative numbers
!SB	Integer	Converts a byte to decimal notation with negative numbers converted properly
!SW	Integer	Converts a word to decimal notation with negative numbers converted properly
!SL	Integer	Converts a longword to decimal notation with negative numbers converted properly
Special formatting:		
!/	None	Inserts a carriage return and a line feed
!_	None	Inserts a tab
!^	None	Inserts a form feed
!!	None	Inserts an exclamation mark
!%I	Integer	Converts a longword integer to a named UIC in the format [group-identifier,member-identifier]
!%S	None	Inserts an "s" if the most recently converted number is not 1 (Not recommended for use with multilingual products.)
!%U	Integer	Converts a longword integer to a numeric UIC in the format [g,m], where g is the group number and m is the member number The directive inserts the brackets and the comma
!n <...!>	None	Left-justifies and blank-fills all data represented by the instructions ... in fields n characters wide
!n*c	None	Repeats the character represented by c for n times
!%T	Integer equal to 0	Inserts the current time
!%D)	Integer equal to 0	Inserts the current date/time

Table DCL–4 (Cont.) Summary of FAO Directives

Directive	Argument Type	Description
Argument interpretation:		
!-	None	Reuses the last argument
!+	None	Skips the next argument

Output Strings from Character String Insertion

The !AS directive inserts a character string (specified as an argument for the directive) into the control string. The field length of the character string when it is inserted into the control string defaults to the length of the character string. If the default length is shorter than an explicitly stated field length, the string is left-justified and blank-filled. If the default length is longer than an explicitly stated field length, the string is truncated on the right.

Output Strings from Zero-Filled Numeric Conversion

Directives for zero-filled numeric conversion convert an integer (specified as an argument for the directive) to decimal, octal, or hexadecimal notation. The ASCII representation of the integer is inserted into the control string. Default output field lengths for the converted argument are determined as follows.

Directives that convert arguments to octal notation return 3 digits for byte conversion, 6 digits for word conversion, and 11 digits for longword conversion. Numbers are right-justified and zero-filled on the left. Explicit-length fields longer than the default are blank-filled on the left. Explicit-length fields shorter than the default are truncated on the left.

Directives that convert arguments to hexadecimal notation return 2 digits for byte conversion, 4 digits for word conversion, and 8 digits for longword conversion. Numbers are right-justified and zero-filled on the left. Explicit-length fields longer than the default are blank-filled on the left. Explicit-length fields shorter than the default are truncated on the left.

Directives that convert arguments to decimal notation return the required number of characters for the decimal number. Explicit-length fields longer than the default are zero-filled on the left. If an explicit-length field is shorter than the number of characters required for the decimal number, the output field is completely filled with asterisks (*).

For byte conversion, only the low-order 8 bits of the binary representation of the argument are used. For word conversion, only the low-order 16 bits of the binary representation of the argument are used. For longword conversion, the entire 32-bit binary representation of the argument is used.

Output Strings from Blank-Filled Numeric Conversion

Directives for blank-filled numeric conversion convert an integer (specified as an argument for the directive) to decimal notation. These directives can convert the integer as a signed or unsigned number. The ASCII representation of the integer is inserted into the control string.

Output field lengths for the converted argument default to the required number of characters. Values shorter than explicit-length fields are right-justified and blank-filled; values longer than explicit-length fields cause the field to be filled with asterisks.

For byte conversion, only the low-order 8 bits of the binary representation of the argument are used. For word conversion, only the low-order 16 bits of the binary representation of the argument are used. For longword conversion, the entire 32-bit binary representation of the argument is used.

EXAMPLES

1

```
$ COUNT = 57
  REPORT = F$FAO("NUMBER OF FORMS = !SL",COUNT)
$ SHOW SYMBOL REPORT
  REPORT = "NUMBER OF FORMS = 57"
```

In this command procedure, the FAO directive !SL is used in a control string to convert the number equated to the symbol COUNT to a character string. The converted string is inserted into the control string.

Note that COUNT is assigned an integer value of 57. The F$FAO function returns the ASCII string, "NUMBER OF FORMS = 57", and assigns the string to the symbol REPORT.

2

```
$ A = "ERR"
$ B = "IS"
$ C = "HUM"
$ D = "AN"
  PHRASE = F$FAO("TO !3(AS)",A,B,C+D)
$ SHOW SYMBOL PHRASE
  PHRASE = "TO ERRISHUMAN"
```

In this command procedure, the !AS directive is used to insert the values assigned to the symbols A, B, C, and D into the control string.

Because the specified repeat count for the !AS directive is 3, F$FAO looks for three arguments. The arguments in this example include the symbol A ("ERR"), the symbol B ("IS"), and the expression C+D ("HUMAN"). Note that the values of these string arguments are concatenated to form the string "ERRISHUMAN".

3

```
$ A = "ERR"
$ B = "IS"
$ C = "HUMAN"
  PHRASE = F$FAO("TO !#(#AS)",3,6,A,B,C)
$ SHOW SYMBOL PHRASE
  PHRASE = "TO ERR   IS    HUMAN "
```

In this command procedure, the F$FAO function is used with the !AS directive to format a character string. The first number sign (#) represents the repeat count given by the first argument, 3. The second number sign (#) represents the field size given by the second argument, 6. The next three arguments (A,B,C) provide the strings that are placed into the control string each time the !AS directive is repeated.

Each argument string is output to a field having a length of six characters. Because each string is less than six characters, each field is left-justified and padded with blank spaces. The resulting string is assigned to the symbol PHRASE.

F$FILE_ATTRIBUTES

Returns attribute information for a specified file.

FORMAT

F$FILE_ATTRIBUTES*(file-spec,item)*

return value

Either an integer or a character string, depending on the item you request. Table DCL–5 shows the data types of the values returned for each item.

ARGUMENTS

file-spec

Specifies the name of the file, as a character string, about which you are requesting information. Only one file name may be specified. Wildcard characters are not allowed.

item

Indicates which attribute of the file is to be returned. The item argument must be specified as a character string expression, and can be any one of the VMS RMS field names listed in Table DCL–5.

DESCRIPTION

Use the F$FILE_ATTRIBUTES lexical function in DCL assignment statements and expressions to return file attribute information. Table DCL–5 lists the items you can specify with the F$FILE_ATTRIBUTES function, the information returned, and the data type of this information.

Table DCL–5 F$FILE_ATTRIBUTES Items

Item	Return Type	Information Returned
ALQ	Integer	Allocation Quantity
BDT	String	Backup Date/Time
BKS	Integer	Bucket Size
BLS	Integer	Block Size
CBT	String	"TRUE" If Contiguous-Best-Try; returns "TRUE" or "FALSE"
CDT	String	Creation Date/Time
CTG	String	"TRUE" If Contiguous; returns "TRUE" or "FALSE"
DEQ	Integer	Default Extension Quantity
DID	String	Directory ID String
DVI	String	Device Name String
EDT	String	Expiration Date/Time
EOF	Integer	Number of Blocks Used
FID	String	File ID String

Table DCL–5 (Cont.) F$FILE_ATTRIBUTES Items

Item	Return Type	Information Returned
FSZ	Integer	Fixed Control Area Size
GRP	Integer	Owner Group Number
KNOWN	String	Known File; returns "TRUE" or "FALSE" to indicate whether file is installed with the Install Utility
MBM	Integer	Owner Member Number
MRN	Integer	Maximum Record Number
MRS	Integer	Maximum Record Size
NOA	Integer	Number of Areas
NOK	Integer	Number of Keys
ORG	String	File Organization; returns "SEQ", "REL", "IDX"
PRO	String	File Protection String
PVN	Integer	Prolog Version Number
RAT	String	Record Attributes; returns "CR", "PRN", "FTN", ""
RCK	String	TRUE If Read Check; returns "TRUE", "FALSE"
RDT	String	Revision Date/Time
RFM	String	Record Format String; returns the values "VAR", "FIX", "VFC", "UDF", "STM", "STMLF","STMCR"
RVN	Integer	Revision Number
UIC	String	Owner UIC String
WCK	String	True If Write Check; returns "TRUE","FALSE"

File attributes are stored in the file header, which is created from information in VMS RMS control blocks. For more information on VMS RMS control blocks, see the *VMS Record Management Services Manual.*

EXAMPLES

1
```
$ FILE_ORG = F$FILE_ATTRIBUTES("QUEST.DAT","ORG")
$ SHOW SYMBOL FILE_ORG
  FILE_ORG = "SEQ"
```

This example uses the F$FILE_ATTRIBUTES function to assign the value of the file organization type to the symbol FILE_ORG. The F$FILE_ATTRIBUTES function returns the character string "SEQ" to show that QUEST.DAT is a sequential file.

The QUEST.DAT and ORG arguments for the F$FILE_ATTRIBUTES function are string literals and must be enclosed in quotation marks when used in expressions.

```
2   $ RFM = F$FILE_ATTRIBUTES("KANSAS::USE$:[CARS]SALES.CMD","RFM")
    $ SHOW SYMBOL RFM
      RFM = "VAR"
```

This example uses the F$FILE_ATTRIBUTES function to return information about a file on a remote node. The function returns the record format string VAR, indicating that records are variable length.

F$GETDVI

Invokes the $GETDVI system service to return a specified item of information for a specified device. This lexical function allows a process to obtain information for a device to which the process has not necessarily assigned a channel.

FORMAT

F$GETDVI*(device-name,item)*

return value

Either an integer or a character string, depending on the item you request. Table DCL–6 shows the data types of the values returned for each item.

ARGUMENTS

device-name

Specifies a physical device name or a logical name equated to a physical device name. Specify the device name as a character string expression.

After the device-name expression is evaluated, the F$GETDVI function examines the first character of the name. If the first character is an underscore (_), the name is considered a physical device name. Otherwise, a single level of logical name translation is performed and the equivalence name, if any, is used.

item

Specifies the type of device information to be returned. The item argument must be specified as a character string expression and may be any one of the items listed in Table DCL–6.

DESCRIPTION

The F$GETDVI function returns information on all items that can be specified with the $GETDVI system service. In addition to the items that the $GETDVI system service allows, the F$GETDVI function allows you to specify the item EXISTS.

Table DCL–6 lists the items you can specify with the F$GETDVI function, the type of information returned, and the data types of the return values. For more information on the $GETDVI system service and the items you can specify, see the *VMS System Services Reference Manual.*

Table DCL–6 F$GETDVI Items

Item	Return Type	Information Returned
ACPPID	String	ACP process ID.
ACPTYPE	String	ACP type code, as one of the following strings: "F11CV1", "F11V2", "JNL", "MTA", "NET", or "REM".
ALL	String	"TRUE" or "FALSE" to indicate whether the device is allocated.
ALLDEVNAM	String	Allocation class device name.

Table DCL–6 (Cont.) F$GETDVI Items

Item	Return Type	Information Returned
ALLOCLASS	Longword integer between 0 and 255	Allocation class of the host.
ALT_HOST_AVAIL	String	"TRUE" or "FALSE" to indicate whether the host serving the alternate path is available.
ALT_HOST_NAME	String	Name of the host serving the alternate path.
ALT_HOST_TYPE	String	Hardware type of the host serving the alternate path.
AVL	String	"TRUE" or "FALSE" to indicate whether the device is available for use.
CCL	String	"TRUE" or "FALSE" to indicate whether the device is a carriage control device.
CLUSTER	Integer	Volume cluster size.
CONCEALED	String	"TRUE" or "FALSE" to indicate whether the logical device name translates to a concealed device.
CYLINDERS	Integer	Number of cylinders on the volume (disk).
DEVBUFSIZ	Integer	Device buffer size.
DEVCHAR	Integer	Device characteristics.
DEVCHAR2	Integer	Additional device characteristics.
DEVCLASS	Integer	Device class. See Table DCL–7 for a list of the values returned.
DEVDEPEND	Integer	Device-dependent information.
DEVDEPEND2	Integer	Additional device-dependent information.
DEVLOCKNAM	String	A unique lock name for the device.
DEVNAM	String	Device name.
DEVSTS	Integer	Device-dependent status information.
DEVTYPE	Integer	Device type. See Table DCL–8 for a list of the values returned.
DIR	String	"TRUE" or "FALSE" to indicate whether the device is directory structured.
DMT	String	"TRUE" or "FALSE" to indicate whether the device is marked for dismount.
DUA	String	"TRUE" or "FALSE" to indicate whether the device is a generic device.
ELG	String	"TRUE" or "FALSE" to indicate whether the device has error logging enabled.
ERRCNT	Integer	Error count.
EXISTS	String	"TRUE" or "FALSE" to indicate whether the device exists on the system.
FOD	String	"TRUE" or "FALSE" to indicate whether the device is a files-oriented device.
FOR	String	"TRUE" or "FALSE" to indicate whether the device is mounted foreign.
FREEBLOCKS	Integer	Number of free blocks on the volume (disk).
FULLDEVNAM	String	Fully qualified device name.

Table DCL–6 (Cont.) F$GETDVI Items

Item	Return Type	Information Returned
GEN	String	"TRUE" or "FALSE" to indicate whether the device is a generic device.
HOST_AVAIL	String	"TRUE" or "FALSE" to indicate whether the host serving the primary path is available.
HOST_COUNT	Integer	Number of hosts that make the device available to other nodes in the VAXcluster.
HOST_NAME	String	Name of the host serving the primary path.
HOST_TYPE	String	Hardware type of the host serving the primary path.
IDV	String	"TRUE" or "FALSE" to indicate whether the device is capable of providing input.
LOCKID	Integer	Cluster-wide lock identification.
LOGVOLNAM	String	Logical volume name.
MAXBLOCK	Integer	Number of logical blocks on the volume.
MAXFILES	Integer	Maximum number of files on the volume. This item code is applicable only to disks.
MBX	String	"TRUE" or "FALSE" to indicate whether the device is a mailbox.
MEDIA_ID	String	Nondecoded media ID.
MEDIA_NAME	String	Either the name of the disk or the tape type.
MEDIA_TYPE	String	Device name prefix.
MNT	String	"TRUE" or "FALSE" to indicate whether the device is mounted.
MOUNTCNT	Integer	Mount count.
NET	String	"TRUE" or "FALSE" to indicate whether the device is a network device.
NEXTDEVNAM	String	Device name of the next volume in a volume set. This item applies only to disks.
ODV	String	"TRUE" or "FALSE" to indicate whether the device is capable of providing output.
OPCNT	Integer	Operation count.
OPR	String	"TRUE" or "FALSE" to indicate whether the device is an operator.
OWNUIC	String	UIC of the device owner.
PID	String	Process identification of the device owner.
RCK	String	"TRUE" or "FALSE" to indicate whether the device has read checking enabled.
RCT	String	"TRUE" or "FALSE" to indicate whether the disk contains RCT.
REC	String	"TRUE" or "FALSE" to indicate whether the device is record oriented.
RECSIZ	Integer	Blocked record size.
REFCNT	Integer	Reference count of processes using the device.
REMOTE_DEVICE	String	"TRUE" or "FALSE" to indicate whether the device is a remote device.
RND	String	"TRUE" or "FALSE" to indicate whether the device allows random access.
ROOTDEVNAM	String	Device name of the root volume in a volume set. This item applies only to disks.

Table DCL–6 (Cont.) F$GETDVI Items

Item	Return Type	Information Returned
RTM	String	"TRUE" or "FALSE" to indicate whether the device is real-time.
SDI	String	"TRUE" or "FALSE" to indicate whether the device is single-directory structured.
SECTORS	Integer	Number of sectors per track. This item applies only to disks.
SERIALNUM	Integer	Volume serial number. This item applies only to disks.
SERVED_DEVICE	String	"TRUE" or "FALSE" to indicate whether the device is a served device.
SHR	String	"TRUE" or "FALSE" to indicate whether the device is shareable.
SPL	String	"TRUE" or "FALSE" to indicate whether the device is being spooled.
SPLDEVNAM	String	Name of the device being spooled.
SQD	String	"TRUE" or "FALSE" to indicate whether the device is sequential block-oriented (that is, magnetic tape).
STS	Integer	Status information.
SWL	String	"TRUE" or "FALSE" to indicate whether the device is software write-locked.
TRACKS	Integer	Number of tracks per cylinder. This item applies only to disks.
TRANSCNT	Integer	Volume transaction count.
TRM	String	"TRUE" or "FALSE" to indicate whether the device is a terminal.
TT_ACCPORNAM	String	The terminal server name and port name.
TT_ALTYPEAHD	String	"TRUE" or "FALSE" to indicate whether the terminal has an alternate type-ahead buffer (terminals only).
TT_ANSICRT	String	"TRUE" or "FALSE" to indicate whether the terminal is an ANSI CRT terminal (terminals only).
TT_APP_KEYPAD	String	"TRUE" or "FALSE" to indicate whether the keypad is in applications mode (terminals only).
TT_AUTOBAUD	String	"TRUE" or "FALSE" to indicate whether the terminal has automatic baud rate detection (terminals only).
TT_AVO	String	"TRUE" or "FALSE" to indicate whether the terminal has a VT100-family terminal display (terminals only).
TT_BLOCK	String	"TRUE" or "FALSE" to indicate whether the terminal has block mode capability (terminals only).
TT_BRDCSTMBX	String	"TRUE" or "FALSE" to indicate whether the terminal uses mailbox broadcast messages (terminals only).
TT_CRFILL	String	"TRUE" or "FALSE" to indicate whether the terminal requires fill after RET (terminals only).
TT_DECCRT	String	"TRUE" or "FALSE" to indicate whether the terminal is a DIGITAL CRT terminal (terminals only).
TT_DECCRT2	String	"TRUE" or "FALSE" to indicate whether the terminal is a DIGITAL CRT2 terminal (terminals only).
TT_DIALUP	String	"TRUE" or "FALSE" to indicate whether the terminal is connected to dialup (terminals only).

Table DCL–6 (Cont.) F$GETDVI Items

Item	Return Type	Information Returned
TT_DISCONNECT	String	"TRUE" or "FALSE" to indicate whether the terminal can be disconnected (terminals only).
TT_DMA	String	"TRUE" or "FALSE" to indicate whether the terminal has DMA mode (terminals only).
TT_DRCS	String	"TRUE" or "FALSE" to indicate whether the terminal supports loadable character fonts (terminals only).
TT_EDIT	String	"TRUE" or "FALSE" to indicate whether the edit characteristic is set.
TT_EDITING	String	"TRUE" or "FALSE" to indicate whether advanced editing is enabled (terminals only).
TT_EIGHTBIT	String	"TRUE" or "FALSE" to indicate whether the terminal uses the 8-bit ASCII character set (terminals only).
TT_ESCAPE	String	"TRUE" or "FALSE" to indicate whether the terminal generates escape sequences (terminals only).
TT_FALLBACK	String	"TRUE" or "FALSE" to indicate whether the terminal uses the multinational fallback option (terminals only).
TT_HALFDUP	String	"TRUE" or "FALSE" to indicate whether the terminal is in half-duplex mode (terminals only).
TT_HANGUP	String	"TRUE" or "FALSE" to indicate whether the hangup characteristic is set (terminals only).
TT_HOSTSYNC	String	"TRUE" or "FALSE" to indicate whether the terminal has host/terminal communication (terminals only).
TT_INSERT	String	"TRUE" or "FALSE" to indicate whether insert-mode is the default line editing mode (terminals only).
TT_LFFILL	String	"TRUE" or "FALSE" to indicate whether the terminal requires fill after LF (terminals only).
TT_LOCALECHO	String	"TRUE" or "FALSE" to indicate whether the local echo characteristic is set (terminals only).
TT_LOWER	String	"TRUE" or "FALSE" to indicate whether the terminal has the lowercase characters set.
TT_MBXDSABL	String	"TRUE" or "FALSE" to indicate whether mailboxes associated with the terminal will receive unsolicited input notification or input notification (terminals only).
TT_MECHFORM	String	"TRUE" or "FALSE" to indicate whether the terminal has mechanical form feed (terminals only).
TT_MECHTAB	String	"TRUE" or "FALSE" to indicate whether the terminal has mechanical tabs and is capable of tab expansion (terminals only).
TT_MODEM	String	"TRUE" or "FALSE" to indicate whether the terminal is connected to a modem (terminals only).
TT_MODHANGUP	String	"TRUE" or "FALSE" to indicate whether the modify hang-up characteristic is set (terminals only).
TT_NOBRDCST	String	"TRUE" or "FALSE" to indicate whether the terminal will receive broadcast messages (terminals only).

Table DCL–6 (Cont.) F$GETDVI Items

Item	Return Type	Information Returned
TT_NOECHO	String	"TRUE" or "FALSE" to indicate whether the input characters are echoed.
TT_NOTYPEAHD	String	"TRUE" or "FALSE" to indicate whether data must be solicited by a read operation.
TT_OPER	String	"TRUE" or "FALSE" to indicate whether the terminal is an operator terminal (terminals only).
TT_PAGE	Integer	Terminal page length (terminals only).
TT_PASTHRU	String	"TRUE" or "FALSE" to indicate whether there is passall with flow control (terminals only).
TT_PHYDEVNAM	String	Physical device name associated with a channel number or virtual terminal.
TT_PRINTER	String	"TRUE" or "FALSE" to indicate whether there is a printer port available (terminals only).
TT_READSYNC	String	"TRUE" or "FALSE" to indicate whether the terminal has read synchronization (terminals only).
TT_REGIS	String	"TRUE" or "FALSE" to indicate whether the terminal has REGIS graphics (terminals only).
TT_REMOTE	String	"TRUE" or "FALSE" to indicate whether the terminal has established modem control (terminals only).
TT_SCOPE	String	"TRUE" or "FALSE" to indicate whether the terminal is a video screen display (terminals only).
TT_SECURE	String	"TRUE" or "FALSE" to indicate whether the terminal can recognize the secure server (terminals only).
TT_SETSPEED	String	"TRUE" or "FALSE" to indicate whether you can set the speed on the terminal line (terminals only).
TT_SIXEL	String	"TRUE" or "FALSE" to indicate whether the sixel is supported (terminals only).
TT_TTSYNC	String	"TRUE" or "FALSE" to indicate whether there is terminal/host synchronization (terminals only).
TT_SYSPWD	String	"TRUE" or "FALSE" to indicate whether the system password is enabled for a particular terminal.
TT_WRAP	String	"TRUE" or "FALSE" to indicate whether a new line should be inserted if the cursor moves beyond the right margin.
UNIT	Integer	The unit number.
VOLCOUNT	Integer	The count of volumes in a volume set. This item applies only to disks.
VOLNAM	String	The volume name.
VOLNUMBER	Integer	Number of the current volume in a volume set. This item applies only to disks.
VOLSETMEM	String	"TRUE" or "FALSE" to indicate whether the device is a volume set (disks only).
VPRO	String	The volume protection mask.
WCK	String	"TRUE" or "FALSE" to indicate whether the device has write checking enabled.

Table DCL–7 Values Returned by the DEVCLASS Item

Device Class	Value	Symbolic Name
Disk device	1	DC$_DISK
Tape device	2	DC$_TAPE
Synchronous Communications device	32	DC$_SCOM
Card reader	65	DC$_CARD
Terminal	66	DC$_TERM
Line printer	67	DC$_LP
Real-time	96	DC$_REALTIME
Bus	128	DC$_BUS
Mailbox	160	DC$_ MAILBOX
Journal	161	DC$_JOURNAL
Miscellaneous device	200	DC$_MISC

Table DCL–8 Values Returned by the DEVTYPE Item

Device Type	Value	Device Type	Value
RK06	1	VT102	98
RK07	2	VT105	99
RP04	3	VT125	100
RP05	4	VT131	101
RP06	5	VT132	102
RM03	6	DZ11	66
RP07	7	DZ32	67
RP07HT	8	DZ730	68
RL01	9	DMC11	1
RL02	10	DMR11	2
RX02	11	XK_3271	3
RX04	12	XJ_2780	4
RX33	36	NW_X25	5
CRX50	33	NV_X29	6
RM80	13	SB_ISB11	7

Table DCL–8 (Cont.) Values Returned by the DEVTYPE Item

Device Type	Value	Device Type	Value
TU58	14	MX_MUX200	8
RM05	15	DMP11	9
RX01	16	DMF32	10
ML11	17	XV_3271	11
RB02	18	CI	12
RB80	19	NI	13
RA80	20	UNA11	14
RA81	21	YN_X25	15
RA60	224	YO_X25	16
RZ01	23	YP_ADCCP	17
RZF01	2	YQ_3271	18
RD51	25	YR_DDCMP	19
RX50	26	YS_SDLC	20
TE16	1	LP11	1
TK50	10	LA11	2
TK60	17	LA180	3
TK70	15	CR11	1
TU45	2	MBX	1
TU77	3	SHRMBX	2
TS11	4	NULL	3
TU78	5	LPA11	1
TA78	6	DR780	2
TU80	7	DR750	3
TU81	8	DR11W	4
TA81	9	PCL11R	5
TTYUNKN	0	PCL11T	6
VT105	1	DR11C	7
FT1	16	XI_DR11C	8
FT2	17	XP_PCL11B	9
FT3	18	CI780	1
FT4	19	C1750	2
FT5	20	UQPORT	3
FT6	21	UDA50	3
FT7	22	UDA50A	4
FT8	23	RC25	23
LAX	32	TU81P	6
LA36	32	RDRX	7
LA120	33	UNKNJNL	0

Table DCL–8 (Cont.) Values Returned by the DEVTYPE Item

Device Type	Value	Device Type	Value
VT5X	64	RUJNL	1
VT52	64	BIJNL	2
VT55	65	AIJNL	3
VT100	96	ATJNL	4
VT101	97	CLJNL	5
VK100	2	DN11	1
VT173	3	VT200	110
LA34	34	LA210	40
LA38	35	LA100	37
LA12	36	LQP02	38
LA24	37		

EXAMPLE

```
$ ERR = F$GETDVI("_DQA0","ERRCNT")
$ SHOW SYMBOL ERR
  ERR = 0  Hex = 00000000 Octal = 000000
```

This example shows how to use the F$GETDVI function to return an error count for the device DQA0. You must place quotation marks around the device-name _DQA0 and the item ERRCNT because they are string literals.

F$GETJPI

Invokes the $GETJPI system service to return accounting, status, and identification information on the specified process.

Requires GROUP privilege to obtain information on other processes in the same group. Requires WORLD privilege to obtain information on any other processes in the system.

FORMAT

F$GETJPI*(pid,item)*

return value

Either an integer or a character string, depending on the item you request. Table DCL–9 shows the data types of the values returned for each item.

ARGUMENTS

pid

Specifies the identification number of the process for which information is being reported. Specify the pid argument as a character string expression. You can omit the leading zeros.

If you specify a null string (""), the current process identification number is used.

You cannot use a wildcard to specify the pid argument in the F$GETJPI function, as you can with the $GETJPI system service. To get a list of process identification numbers, use the F$PID function.

item

Indicates the type of process information to be returned. *Item* must be specified as a character string expression and may be any one of the items listed in Table DCL–9.

DESCRIPTION

The F$GETJPI function returns information on all items that can be specified with the $GETJPI system service. For more information on the $GETJPI system service, see the *VMS System Services Reference Manual.*

Table DCL–9 lists the items you can specify with the F$GETJPI function, the information returned, and the data type of this information.

If you use the F$GETJPI function to request information on the null process or the swapper process, you can specify any of the items in Table DCL–9 except: ACCOUNT, BYTLM, ENQCNT, ENQLM, EXCVEC, FILCNT, FILM, FINALEXC, IMAGNAME, LOGINTIM, MSGMASK, PAGFILCNT, PGFLQUOTA, PRCCNT, PRCLM, PROCPRIV, SITESPEC, TQCNT, TQLM, USERNAME, VIRTPEAK, VOLUMES, WSPEAK.

Table DCL–9 F$GETJPI Items

Item	Return Type	Information Returned
ACCOUNT	String	Account name string (8 characters filled with trailing blanks)
APTCNT	Integer	Active page table count
ASTACT	Integer	Access modes with active ASTs
ASTCNT	Integer	Remaining AST quota
ASTEN	Integer	Access modes with ASTs enabled
ASTLM	Integer	AST limit quota
AUTHPR	Integer	Maximum priority that a process without the ALTPRI privilege can achieve with the $SETPRI system service
AUTHPRIV	String	Privileges that a process is authorized to enable
BIOCNT	Integer	Remaining buffered I/O quota
BIOLM	Integer	Buffered I/O limit quota
BUFIO	Integer	Count of process buffered I/O operations
BYTCNT	Integer	Remaining buffered I/O byte count quota
BYTLM	Integer	Buffered I/O byte count limit quota
CLINAME	String	Current command language interpreter; always returns "DCL"
CPULIM	Integer	Limit on process CPU time
CPUTIM	Integer	CPU time used in hundredths of a second
CURPRIV	String	Current process privileges
DFPFC	Integer	Default page fault cluster size
DFWSCNT	Integer	Default working set size
DIOCNT	Integer	Remaining direct I/O quota
DIOLM	Integer	Direct I/O limit quota
DIRIO	Integer	Count of direct I/O operations for the process
EFCS	Integer	Local event flags 0 through 31
EFCU	Integer	Local event flags 32 through 63
EFWM	Integer	Event flag wait mask
ENQCNT	Integer	Lock request quota remaining
ENQLM	Integer	Lock request quota limit
EXCVEC	Integer	Address of a list of exception vectors
FILCNT	Integer	Remaining open file quota
FILLM	Integer	Open file quota
FINALEXC	Integer	Address of a list of final exception vectors
FREP0VA	Integer	First free page at end of program region (P0 space)
FREP1VA	Integer	First free page at end of control region (P1 space)

Table DCL–9 (Cont.) F$GETJPI Items

Item	Return Type	Information Returned
FREPTECNT	Integer	Number of pages available for virtual memory expansion
GPGCNT	Integer	Global page count in working set
GRP	Integer	Group number of the UIC
IMAGECOUNT	Integer	Number of images that have been run down for the process
IMAGNAME	String	File name of the current image
IMAGPRIV	String	Privileges with which the current image was installed
JOBPRCCNT	Integer	Number of subprocesses owned by the process
LOGINTIM	String	Process creation time
MASTER_PID	String	Returns the process identification of the process at the top of the current job's process tree
MEM	Integer	Member number of the UIC
MODE	String	Current process mode ("BATCH", "INTERACTIVE", "NETWORK", or "OTHER")
MSGMASK	Integer	Default message mask
OWNER	String	Process identification number of process owner
PAGEFLTS	Integer	Count of page faults
PAGFILCNT	Integer	Remaining paging file quota
PAGFILLOC	Integer	Location of the paging file
PGFLQUOTA	Integer	Paging file quota (maximum virtual page count)
PHDFLAGS	Integer	Flags word
PID	String	Process identification number
PPGCNT	Integer	Process page count
PRCCNT	Integer	Count of subprocesses
PRCLM	Integer	Subprocess quota
PRCNAM	String	Process name
PRIB	Integer	Process's base priority
PROCPRIV	Integer	Process's default privileges
SITESPEC	Integer	Per-process site-specific longword
STATE	String	Process state
STS	Integer	Process status flags
SWPFILLOC	Integer	Location of the swap file
TERMINAL	String	Login terminal name for interactive users (1-7 characters)
TMBU	Integer	Termination mailbox unit number
TQCNT	Integer	Remaining timer queue entry quota
TQLM	Integer	Timer queue entry quota

Table DCL–9 (Cont.) F$GETJPI Items

Item	Return Type	Information Returned
UIC	String	Process's UIC
USERNAME	String	User name string (12 characters filled with trailing blanks)
VIRTPEAK	Integer	Peak virtual address size
VOLUMES	Integer	Count of currently mounted volumes
WSAUTH	Integer	Maximum authorized working set size
WSAUTHEXT	Integer	Maximum authorized working set extent
WSEXTENT	Integer	Current working set extent
WSPEAK	Integer	Working set peak
WSQUOTA	Integer	Working set size quota
WSSIZE	Integer	Process's current working set size

EXAMPLE

```
$ NAME = F$GETJPI("3B0018","USERNAME")
$ SHOW SYMBOL NAME
  NAME = "JANE        "
```

This example shows how to use the F$GETJPI function to return the username for the process number 3B0018. The username is assigned to the symbol NAME.

F$GETQUI

Invokes the $GETQUI system service to return information about queues, batch and print jobs currently in those queues, form definitions, and characteristic definitions kept in the system job queue file.

Requires READ access to the job or SYSPRV or OPER privilege to obtain job and file information.

FORMAT

F$GETQUI*(function,[item],[object-id],[flags])*

return value

Either an integer or a character string, depending on the item you request. For items that return a Boolean value, the string is "TRUE" or "FALSE."

ARGUMENTS

function

Specifies the action that the F$GETQUI lexical function is to perform. F$GETQUI supports all functions that can be specified with the $GETQUI system service.

Function	Description
CANCEL_OPERATION	Terminates any wildcard operation that may have been initiated by a previous call to F$GETQUI.
DISPLAY_CHARACTERISTIC	Returns information about a specific characteristic definition or the next characteristic definition in a wildcard operation.
DISPLAY_ENTRY	Returns information about a specific job entry or the next job entry that matches the selection criteria in a wildcard operation. The DISPLAY_ENTRY function code is similar to the DISPLAY_JOB function code in that both return job information. DISPLAY_JOB, however, requires that a call be made to establish queue context; DISPLAY_ENTRY does not require that queue context be established.
DISPLAY_FILE	Returns information about the next file defined for the current job context. Before you make a call to F$GETQUI to request file information, you must make a call to display queue and job information (with the DISPLAY_QUEUE and DISPLAY_JOB function codes) or display entry information (with the DISPLAY_ENTRY function code).
DISPLAY_FORM	Returns information about a specific form definition or the next form definition in a wildcard operation.
DISPLAY_JOB	Returns information about the next job defined for the current queue context. Before you make a call to F$GETQUI to request job information, you must make a call to display queue information (with the DISPLAY_QUEUE function code). The DISPLAY_JOB function code is similar to the DISPLAY_ENTRY function code in that both return job information. DISPLAY_JOB, however, requires that a call be made to establish queue context; DISPLAY_ENTRY does not require that queue context be established.

Function	Description
DISPLAY_QUEUE	Returns information about a specific queue definition or the next queue definition in a wildcard operation.
TRANSLATE_QUEUE	Translates a logical name for a queue to the equivalence name for the queue.

Some function arguments cannot be specified with the item-code, object-id, or flags argument. The following table lists each function argument and corresponding format line to show whether the item-code, object-id, and flags arguments are required, optional, or not applicable for that specific function. In the following format lines, brackets denote an optional argument. An omitted argument means the argument is not applicable for that function. Note that two commas must be used as placeholders to denote an omitted (whether optional or not applicable) argument.

Function	Format Line
CANCEL_OPERATION	F$GETQUI("CANCEL_OPERATION") or F$GETQUI("")
DISPLAY_CHARACTERISTIC	F$GETQUI("DISPLAY_CHARACTERISTIC",[item],object-id,[flags])
DISPLAY_ENTRY	F$GETQUI("DISPLAY_ENTRY",[item],[object-id],[flags])
DISPLAY_FILE	F$GETQUI("DISPLAY_FILE",[item],,[flags])
DISPLAY_FORM	F$GETQUI("DISPLAY_FORM",[item],object-id,[flags])
DISPLAY_JOB	F$GETQUI("DISPLAY_JOB",[item],,[flags])
DISPLAY_QUEUE	F$GETQUI("DISPLAY_QUEUE",[item],object-id,[flags])
TRANSLATE_QUEUE	F$GETQUI("TRANSLATE_QUEUE",[item],object-id)

item

Corresponds to a $GETQUI system service output item code. *Item* specifies the nature of the information you want returned about a particular queue, job, file, form, or characteristic. Table DCL–1 lists each item code and the data type of the value returned for each item code.

object-id

Corresponds to the $GETQUI system service search-name and search-number input item codes. *Object-id* specifies either the name or number of an object (for example, a specific queue name or form number) about which F$GETQUI is to return information. Wildcard names are allowed for the following functions:

- DISPLAY_CHARACTERISTIC
- DISPLAY_ENTRY
- DISPLAY_FORM
- DISPLAY_QUEUE

By specifying a wildcard as the object-id argument on successive calls, you can get status information about one or more jobs in a specific queue or about files within jobs in a specific queue. When a wildcard name is used, each call returns information for the next object (queue, form, and so on) in the list. A null string ("") is returned when the end of the list is reached. A wildcard can represent only object names, not object numbers.

flags

Specifies a list of keywords, separated by commas, that corresponds to the flags defined for the $GETQUI system service search-flags input item code. (These flags are used to define the scope of the object search specified in the call to the $GETQUI system service.) Note that these keywords can be used only with certain function codes:

Keyword	Valid Function Code	Description
ALL_JOBS	DISPLAY_JOB	Requests that F$GETQUI search all jobs included in the established queue context. If you do specify this flag, F$GETQUI returns information only about jobs that have the same user name as the caller.
BATCH	DISPLAY_QUEUE	Selects batch queues.
	DISPLAY_ENTRY	
EXECUTING_JOBS	DISPLAY_ENTRY	Selects executing jobs.
	DISPLAY_JOB	
FREEZE_CONTEXT	DISPLAY_ CHARACTERISTIC	When in wildcard mode, prevents advance of wildcard context to the next object.
	DISPLAY_ENTRY	
	DISPLAY_FILE	
	DISPLAY_FORM	
	DISPLAY_JOB	
	DISPLAY_QUEUE	
GENERIC	DISPLAY_QUEUE	Selects generic queues for searching.
HOLDING_JOBS	DISPLAY_ENTRY	Selects jobs on unconditional hold.
	DISPLAY_JOB	
PENDING_JOBS	DISPLAY_ENTRY	Selects pending jobs.
	DISPLAY_JOB	
PRINTER	DISPLAY_QUEUE	Selects printer queues.
	DISPLAY_ENTRY	
RETAINED_JOBS	DISPLAY_ENTRY	Selects jobs being retained.
	DISPLAY_JOB	
SERVER	DISPLAY_QUEUE	Selects server queues.
	DISPLAY_ENTRY	
SYMBIONT	DISPLAY_QUEUE	Selects all output queues. Equivalent to specifying "PRINTER,SERVER,TERMINAL".
	DISPLAY_ENTRY	
TERMINAL	DISPLAY_QUEUE	Selects terminal queues.
	DISPLAY_ENTRY	
THIS_JOB	DISPLAY_FILE	Selects all job file information about the calling batch job, the command file being executed, or the queue associated with the calling batch job.
	DISPLAY_JOB	

Keyword	Valid Function Code	Description
	DISPLAY_QUEUE	
TIMED_RELEASE_JOBS	DISPLAY_ENTRY	Selects jobs on hold until a specified time.
	DISPLAY_JOB	
WILDCARD	DISPLAY_ CHARACTERISTIC	Establishes and saves a context. Because the context is saved, the next operation can be performed based on that context.
	DISPLAY_ENTRY	
	DISPLAY_FORM	
	DISPLAY_QUEUE	

DESCRIPTION

The F$GETQUI lexical function provides all the features of the $GETQUI system service, including wildcard and nested wildcard operations. See the description of the $GETQUI system service in the *VMS System Services Reference Manual* for more information.

The F$GETQUI function returns information on all items that can be specified with the $GETQUI system service. Table DCL-10 lists the items you can specify with the F$GETQUI function, the information returned, and the data type of this information.

Table DCL–10 F$GETQUI Items

Item	Return Type	Information Returned
ACCOUNT_NAME	String	The account name of the owner of the specified job.
AFTER_TIME	String	The system time at or after which the specified job can execute.
ASSIGNED_QUEUE_NAME	String	The name of the execution queue to which the logical queue specified in the call to F$GETQUI is assigned.
BASE_PRIORITY	Integer	The priority at which batch jobs are initiated from a batch execution queue or the priority of a symbiont process that controls output execution queues.
CHARACTERISTICS	String	The characteristics associated with the specified queue or job.
CHARACTERISTIC_NAME	String	The name of the specified characteristic.
CHARACTERISTIC_ NUMBER	Integer	The number of the specified characteristic.
CHECKPOINT_DATA	String	The value of the DCL symbol BATCH$RESTART when the specified batch job is restarted.
CLI	String	The name of the command language interpreter used to execute the specified batch job. The file specification returned assumes the device name SYS$SYSTEM and the file type EXE.
COMPLETED_BLOCKS	Integer	The number of blocks that the symbiont has processed for the specified print job. This item code is applicable only to print jobs.
CONDITION_VECTOR	Integer	The completion status of the specified job.

Table DCL–10 (Cont.) F$GETQUI Items

Item	Return Type	Information Returned
CPU_DEFAULT	String	The default CPU time limit specified for the queue in 10-millisecond units. This item code is applicable only to batch execution queues.
CPU_LIMIT	String	The maximum CPU time limit specified for the specified job or queue in 10-millisecond units. This item code is applicable only to batch jobs and batch execution queues.
DEFAULT_FORM_NAME	String	The name of the default form associated with the specified output queue.
DEFAULT_FORM_STOCK	String	The name of the paper stock on which the specified default form is to be printed.
DEVICE_NAME	String	The node and device (or both) on which the specified execution queue is located. For batch execution queues, only the node name is returned. For output execution queues, only the device name is returned. The node name is used only in VAXcluster systems. The node name is specified by the SYSGEN parameter SCSNODE for the processor on which the queue executes.
ENTRY_NUMBER	Integer	The queue entry number of the specified job.
EXECUTING_JOB_COUNT	Integer	The number of jobs in the queue that are currently executing.
FILE_BURST	String	"TRUE" or "FALSE" to indicate whether burst and flag pages are to be printed preceding a file.
FILE_CHECKPOINTED	String	"TRUE" or "FALSE" to indicate whether file is checkpointed.
FILE_COPIES	Integer	The number of times the specified file is to be processed. This item code is applicable only to output execution queues.
FILE_COPIES_DONE	Integer	The number of times the specified file has been processed. This item code is applicable only to output execution queues.
FILE_DELETE	String	"TRUE" or "FALSE" to indicate whether file is to be deleted after execution of request.
FILE_DOUBLE_SPACE	String	"TRUE" or "FALSE" to indicate whether the symbiont formats the file with double spacing.
FILE_EXECUTING	String	"TRUE" or "FALSE" to indicate whether file is being processed.
FILE_FLAG	String	"TRUE" or "FALSE" to indicate whether a flag page is to be printed preceding a file.
FILE_FLAGS	Integer	The processing options that have been selected for the specified file. See *VMS System Services Reference Manual* for a list of the available processing options.
FILE_IDENTIFICATION	String	The internal file-identification value that uniquely identifies the selected file. This value specifies (in order) the following three file-identification fields in the RMS NAM block: the 16-byte NAM$T_DVI field, the 6-byte NAM$W_FID field, and the 6-byte NAM$W_DID field.
FILE_PAGE_HEADER	String	"TRUE" or "FALSE" to indicate whether page header is to be printed on each page of output.

Table DCL–10 (Cont.) F$GETQUI Items

Item	Return Type	Information Returned
FILE_PAGINATE	String	"TRUE" or "FALSE" to indicate whether symbiont paginates output by inserting a form feed whenever output reaches the bottom margin of the form.
FILE_PASSALL	String	"TRUE" or "FALSE" to indicate whether symbiont prints the file in PASSALL mode.
FILE_SETUP_MODULES	String	The names of the text modules that are to be extracted from the device control library and copied to the printer before the specified file is printed. This item code is meaningful only for output execution queues.
FILE_SPECIFICATION	String	The fully qualified RMS file specification of the file about which F$GETQUI is returning information.
FILE_STATUS	Integer	File status information. See the *VMS System Services Reference Manual* for a list of the file status conditions.
FILE_TRAILER	String	"TRUE" or "FALSE" to indicate whether trailer page is to be printed following a file.
FIRST_PAGE	Integer	The page number at which the printing of the specified file is to begin. This item code is applicable only to output execution queues.
FORM_DESCRIPTION	String	The text string that describes the specified form to users and operators.
FORM_FLAGS	Integer	The processing options that have been selected for the specified form. See the *VMS System Services Reference Manual* for a list of the available processing options.
FORM_LENGTH	Integer	The physical length of the specified form in lines. This item code is applicable only to output execution queues.
FORM_MARGIN_BOTTOM	Integer	The bottom margin of the specified form in lines.
FORM_MARGIN_LEFT	Integer	The left margin of the specified form in characters.
FORM_MARGIN_RIGHT	Integer	The right margin of the specified form in characters.
FORM_MARGIN_TOP	Integer	The top margin of the specified form in lines.
FORM_NAME	String	The name of the specified form or the mounted form associated with the specified job or queue.
FORM_NUMBER	Integer	The number of the specified form.
FORM_SETUP_MODULES	String	The names of the text modules that are to be extracted from the device control library and copied to the printer before a file is printed on the specified form. This item code is meaningful only for output execution queues.
FORM_SHEET_FEED	String	"TRUE" or "FALSE" to indicate whether symbiont pauses at the end of each physical page so that another sheet of paper can be inserted.
FORM_STOCK	String	The name of the paper stock on which the specified form is to be printed.
FORM_TRUNCATE	String	"TRUE" or "FALSE" to indicate whether printer discards any characters that exceed the specified right margin.

Table DCL–10 (Cont.) F$GETQUI Items

Item	Return Type	Information Returned
FORM_WIDTH	Integer	The width of the specified form.
FORM_WRAP	String	"TRUE" or "FALSE" to indicate whether the printer prints any characters that exceed the specified right margin on the following line.
GENERIC_TARGET	String	The names of the execution queues that are enabled to accept work from the specified generic queue. This item code is meaningful only for generic queues.
HOLDING_JOB_COUNT	Integer	The number of jobs in the queue being held until explicitly released.
INTERVENING_BLOCKS	Integer	The number of blocks to be processed before the specified job can begin to execute. This item code is meaningful only for output execution queues.
INTERVENING_JOBS	Integer	The number of jobs that are to be processed before the specified job can begin to execute. This item code is meaningful only for output execution queues.
JOB_ABORTING	String	"TRUE" or "FALSE" to indicate whether system is attempting to abort execution of job.
JOB_COPIES	Integer	The number of times the specified print job is to be repeated.
JOB_COPIES_DONE	Integer	The number of times that the specified print job has been repeated.
JOB_CPU_LIMIT	String	"TRUE" or "FALSE" to indicate whether a CPU time limit is specified for the job.
JOB_EXECUTING	String	"TRUE" or "FALSE" to indicate whether job is executing or printing.
JOB_FILE_BURST	String	"TRUE" or "FALSE" to indicate whether a burst page option is explicitly specified for the job.
JOB_FILE_BURST_ONE	String	"TRUE" or "FALSE" to indicate whether burst and flag pages precede only the first copy of the first file in the job.
JOB_FILE_FLAG	String	"TRUE" or "FALSE" to indicate whether flag page precedes each file in the job.
JOB_FILE_FLAG_ONE	String	"TRUE" or "FALSE" to indicate whether flag page precedes only the first copy of the first file in the job.
JOB_FILE_PAGINATE	String	"TRUE" or "FALSE" to indicate whether a paginate option is explicitly specified for the job.
JOB_FILE_TRAILER	String	"TRUE" or "FALSE" to indicate whether trailer page follows each file in the job.
JOB_FILE_TRAILER_ONE	String	"TRUE" or "FALSE" to indicate whether trailer page follows only the last copy of the last file in the job.
JOB_FLAGS	Integer	The processing options that have been selected for the specified job.
JOB_HOLDING	String	"TRUE" or "FALSE" to indicate whether job will be held until it is explicitly released.

Table DCL–10 (Cont.) F$GETQUI Items

Item	Return Type	Information Returned
JOB_INACCESSIBLE	String	"TRUE" or "FALSE" to indicate whether caller does not have READ access to the specific job and file information in the system queue file. Therefore, the DISPLAY_JOB and DISPLAY_FILE operations can return information for only the following output value item codes: AFTER_TIME COMPLETED_BLOCKS ENTRY_NUMBER INTERVENING_BLOCKS INTERVENING_JOBS JOB_SIZE JOB_STATUS.
JOB_LIMIT	Integer	The number of jobs that can execute simultaneously on the specified queue. This item code is applicable only to batch execution queues.
JOB_LOG_DELETE	String	"TRUE" or "FALSE" to indicate whether the log file is deleted after it is printed.
JOB_LOG_NULL	String	"TRUE" or "FALSE" to indicate whether no log file is created.
JOB_LOG_SPOOL	String	"TRUE" or "FALSE" to indicate whether job log file is queued for printing when job is complete.
JOB_LOWERCASE	String	"TRUE" or "FALSE" to indicate whether job is to be printed on printer that can print both uppercase and lowercase letters.
JOB_NAME	String	The name of the specified job.
JOB_NOTIFY	String	"TRUE" or "FALSE" to indicate whether message is broadcast to terminal when job completes or aborts.
JOB_PENDING	String	"TRUE" or "FALSE" to indicate whether job is pending.
JOB_PID	String	The process identification (PID) of the executing batch job.
JOB_REFUSED	String	"TRUE" or "FALSE" to indicate whether job was refused by symbiont and is waiting for symbiont to accept it for processing.
JOB_RESET_MODULES	String	The names of the text modules that are to be extracted from the device control library and copied to the printer before each job in the specified queue is printed. This item code is meaningful only for output execution queues.
JOB_RESTART	String	"TRUE" or "FALSE" to indicate whether job will restart after a system failure or can be requeued during execution.
JOB_RETAINED	String	"TRUE" or "FALSE" to indicate whether job has completed, but is being retained in the queue.
JOB_SIZE	Integer	The total number of blocks in the specified print job.
JOB_SIZE_MAXIMUM	Integer	The maximum number of blocks that a print job initiated from the specified queue can contain. This item code is applicable only to output execution queues.
JOB_SIZE_MINIMUM	Integer	The minimum number of blocks that a print job initiated from the specified queue can contain. This item code is applicable only to output execution queues.

Table DCL–10 (Cont.) F$GETQUI Items

Item	Return Type	Information Returned
JOB_STARTING	String	"TRUE" or "FALSE" to indicate whether job controller is starting to process the job and has begun communicating with an output symbiont or a job controller on another node.
JOB_STATUS	Integer	The specified job's status flags. See the *VMS System Services Reference Manual* for a list of the available status flags.
JOB_SUSPENDED	String	"TRUE" or "FALSE" to indicate whether job is suspended.
JOB_TIMED_RELEASE	String	"TRUE" or "FALSE" to indicate whether job is waiting for specified time to execute.
JOB_WSDEFAULT	String	"TRUE" or "FALSE" to indicate whether default working set size is specified for the job.
JOB_WSEXTENT	String	"TRUE" or "FALSE" to indicate whether working set extent is specified for the job.
JOB_WSQUOTA	String	"TRUE" or "FALSE" to indicate whether working set quota is specified for the job.
LAST_PAGE	Integer	The page number at which the printing of the specified file should end. This item code is applicable only to output execution queues.
LIBRARY_SPECIFICATION	String	The name of the device control library for the specified queue. The library specification assumes the device and directory name SYS$LIBRARY and a file type of TLB. This item code is meaningful only for output execution queues.
LOG_QUEUE	String	The name of the queue into which the log file produced for the specified batch job is to be entered for printing. This item code is applicable only to batch jobs.
LOG_SPECIFICATION	String	The name of the log file to be produced for the specified job. This item code is meaningful only for batch jobs.
NOTE	String	The note that is to be printed on the job flag and file flag pages of the specified job. This item code is meaningful only for output execution queues.
OPERATOR_REQUEST	String	The message that is to be sent to the queue operator before the specified job begins to execute. This item code is meaningful only for output execution queues.
OWNER_UIC	String	The owner UIC of the specified queue.
PAGE_SETUP_MODULES	String	The names of the text modules to be extracted from the device control library and copied to the printer before each page of the specified form is printed. This item code is meaningful only for output execution queues.
PARAMETER_1 through PARAMETER_8	String	The value of the user-defined parameters that in batch jobs become the value of the DCL symbol P1 through P8 respectively.
PENDING_JOB_BLOCK_COUNT	Integer	The total number of blocks for all pending jobs in the queue (valid only for output execution queues).
PENDING_JOB_COUNT	Integer	The number of jobs in the queue in a pending state.

Table DCL–10 (Cont.) F$GETQUI Items

Item	Return Type	Information Returned
PENDING_JOB_REASON	Integer	The reason that the job is in a pending state. See the *VMS System Services Reference Manual* for a list of the available flags.
PEND_CHAR_MISMATCH	String	"TRUE" or "FALSE" to indicate whether job requires characteristics that are not available on the execution queue.
PEND_JOB_SIZE_MAX	String	"TRUE" or "FALSE" to indicate whether block size of job exceeds the upper block limit of the execution queue.
PEND_JOB_SIZE_MIN	String	"TRUE" or "FALSE" to indicate whether block size of job is less than the lower limit of the execution queue.
PEND_LOWERCASE_ MISMATCH	String	"TRUE" or "FALSE" to indicate whether job requires lowercase printer.
PEND_NO_ACCESS	String	"TRUE" or "FALSE" to indicate whether owner of job does not have access to the execution queue.
PEND_QUEUE_BUSY	String	"TRUE" or "FALSE" to indicate whether job is pending because the number of jobs currently executing on the queue equals the job limit for the queue.
PEND_QUEUE_STATE	String	"TRUE" or "FALSE" to indicate whether job is pending because the execution queue is not in a running, open state.
PEND_STOCK_MISMATCH	String	"TRUE" or "FALSE" to indicate whether the stock type required by the job's form does not match the stock type of the form mounted on the execution queue.
PRIORITY	Integer	The scheduling priority of the specified job.
PROCESSOR	String	The name of the symbiont image that executes print jobs initiated from the specified queue.
PROTECTION	String	The specified queue's protection mask.
QUEUE_ACL_SPECIFIED	String	"TRUE" or "FALSE" to indicate whether an access control list has been specified for the queue.
QUEUE_ALIGNING	String	"TRUE" or "FALSE" to indicate whether queue prints a specified amount of output so that paper can be properly aligned.
QUEUE_BATCH	String	"TRUE" or "FALSE" to indicate whether queue is a batch queue or a generic batch queue.
QUEUE_CLOSED	String	"TRUE" or "FALSE" to indicate whether queue is closed and will not accept new jobs until the queue is put in an open state.
QUEUE_CPU_DEFAULT	String	"TRUE" or "FALSE" to indicate whether a default CPU time limit has been specified for all jobs in the queue.
QUEUE_CPU_LIMIT	String	"TRUE" or "FALSE" to indicate whether a maximum CPU time limit has been specified for all jobs in the queue.
QUEUE_FILE_BURST	String	"TRUE" or "FALSE" to indicate whether burst and flag pages precede each file in each job initiated from the queue.
QUEUE_FILE_BURST_ONE	String	"TRUE" or "FALSE" to indicate whether burst and flag pages precede only the first copy of the first file in each job initiated from the queue.

Table DCL–10 (Cont.) F$GETQUI Items

Item	Return Type	Information Returned
QUEUE_FILE_FLAG	String	"TRUE" or "FALSE" to indicate whether flag page precedes each file in each job initiated from the queue.
QUEUE_FILE_FLAG_ONE	String	"TRUE" or "FALSE" to indicate whether flag page precedes only the first copy of the first file in each job initiated from the queue.
QUEUE_FILE_PAGINATE	String	"TRUE" or "FALSE" to indicate whether output symbiont paginates output for each job initiated from this queue. The output symbiont paginates output by inserting a form feed whenever output reaches the bottom margin of the form.
QUEUE_FILE_TRAILER	String	"TRUE" or "FALSE" to indicate whether trailer page follows each file in each job initiated from the queue.
QUEUE_FILE_TRAILER_ ONE	String	"TRUE" or "FALSE" to indicate whether trailer page follows only the last copy of the last file in each job initiated from the queue.
QUEUE_FLAGS	Integer	The processing options that have been selected for the specified queue. See the *VMS System Services Reference Manual* for a list of the available processing options.
QUEUE_GENERIC	String	"TRUE" or "FALSE" to indicate whether the queue is a generic queue.
QUEUE_GENERIC_ SELECTION	String	"TRUE" or "FALSE" to indicate whether the queue is an execution queue that can accept work from a generic queue.
QUEUE_IDLE	String	"TRUE" or "FALSE" to indicate whether queue prints a specified amount of output so that paper can be properly aligned.
QUEUE_JOB_BURST	String	"TRUE" or "FALSE" to indicate whether burst and flag pages precede each job initiated from the queue.
QUEUE_JOB_FLAG	String	"TRUE" or "FALSE" to indicate whether a flag page precedes each job initiated from the queue.
QUEUE_JOB_SIZE_SCHED	String	"TRUE" or "FALSE" to indicate whether jobs initiated from the queue are scheduled according to size, with the smallest job of a given priority processed first. (Meaningful only for output queues.)
QUEUE_JOB_TRAILER	String	"TRUE" or "FALSE" to indicate whether a trailer page follows each job initiated from the queue.
QUEUE_LOWERCASE	String	"TRUE" or "FALSE" to indicate whether queue is associated with a printer that can print both uppercase and lowercase characters.
QUEUE_NAME	String	The name of the specified queue or the name of the queue that contains the specified job.
QUEUE_PAUSED	String	"TRUE" or "FALSE" to indicate whether execution of all current jobs in the queue is temporarily halted.
QUEUE_PAUSING	String	"TRUE" or "FALSE" to indicate whether queue is temporarily halting execution. Currently executing jobs are completing; temporarily, no new jobs can begin executing.
QUEUE_PRINTER	String	"TRUE" or "FALSE" to indicate whether the queue is a printer queue.

Table DCL–10 (Cont.) F$GETQUI Items

Item	Return Type	Information Returned
QUEUE_RECORD_BLOCKING	String	"TRUE" or "FALSE" to indicate whether the symbiont is permitted to concatenate, or block together, the output records it sends to the output device.
QUEUE_REMOTE	String	"TRUE" or "FALSE" to indicate whether queue is assigned to a physical device that is not connected to the local node.
QUEUE_RESETTING	String	"TRUE" or "FALSE" to indicate whether queue is resetting and stopping.
QUEUE_RESUMING	String	"TRUE" or "FALSE" to indicate whether queue is restarting after pausing.
QUEUE_RETAIN_ALL	String	"TRUE" or "FALSE" to indicate whether all jobs initiated from the queue remain in the queue after they finish executing. Completed jobs are marked with a completion status.
QUEUE_RETAIN_ERROR	String	"TRUE" or "FALSE" to indicate whether only jobs that do not complete successfully are retained in the queue.
QUEUE_SERVER	String	"TRUE" or "FALSE" to indicate whether queue processing is directed to a server symbiont.
QUEUE_STALLED	String	"TRUE" or "FALSE" to indicate whether physical device to which queue is assigned is stalled; that is, the device has not completed the last I/O request submitted to it.
QUEUE_STARTING	String	"TRUE" or "FALSE" to indicate whether queue is starting.
QUEUE_STATUS	Integer	The specified queue's status flags. See the *VMS System Services Reference Manual* for a list of the available status flags.
QUEUE_STOPPED	String	"TRUE" or "FALSE" to indicate whether queue is stopped.
QUEUE_STOPPING	String	"TRUE" or "FALSE" to indicate whether queue is stopping.
QUEUE_SWAP	String	"TRUE" or "FALSE" to indicate whether jobs initiated from the queue can be swapped.
QUEUE_TERMINAL	String	"TRUE" or "FALSE" to indicate whether the queue is a generic queue that can place jobs only in terminal queues.
QUEUE_UNAVAILABLE	String	"TRUE" or "FALSE" to indicate whether physical device to which queue is assigned is not available.
QUEUE_WSDEFAULT	String	"TRUE" or "FALSE" to indicate whether default working set size is specified for each job initiated from the queue.
QUEUE_WSEXTENT	String	"TRUE" or "FALSE" to indicate whether working set extent is specified for each job initiated from the queue.
QUEUE_WSQUOTA	String	"TRUE" or "FALSE" to indicate whether working set quota is specified for each job initiated from the queue.
REQUEUE_QUEUE_NAME	String	The name of the queue to which the specified job is reassigned.
RESTART_QUEUE_NAME	String	The name of the queue in which the job will be placed if the job is restarted.
RETAINED_JOB_COUNT	Integer	The number of jobs in the queue retained after successful completion plus those retained on error.

Table DCL–10 (Cont.) F$GETQUI Items

Item	Return Type	Information Returned
SCSNODE_NAME	String	The 6-byte name of the VAX node on which jobs initiated from the specified queue execute. The node name matches the value of the SYSGEN parameter SCSNODE for the target node.
SUBMISSION_TIME	String	The time at which the specified job was submitted to the queue.
TIMED_RELEASE_JOB_COUNT	Integer	The number of jobs in the queue on hold until a specified time.
UIC	String	The UIC of the owner of the specified job.
USERNAME	String	The user name of the owner of the specified job.
WSDEFAULT	Integer	The default working set size specified for the specified job or queue. This value is meaningful only for batch jobs and execution and output queues.
WSEXTENT	Integer	The working set extent specified for the specified job or queue. This value is meaningful only for batch jobs and execution and output queues.
WSQUOTA	Integer	The working set quota for the specified job or queue. This value is meaningful only for batch jobs and execution and output queues.

EXAMPLES

1

```
$  BLOCKS = F$GETQUI("DISPLAY_ENTRY", "JOB_SIZE", 1347)
```

In this example, the F$GETQUI lexical function is used to obtain the size in blocks of print job 1347. The value returned reflects the total number of blocks occupied by the files associated with the job.

2

```
$  IF F$GETQUI("DISPLAY_QUEUE", "QUEUE_STOPPED", "VAX1_BATCH") THEN GOTO 500
```

In this example, the F$GETQUI lexical function is used to return a value of "TRUE" or "FALSE" depending on whether the queue VAX1_BATCH is in a stopped state.

3

```
$  TEMP = F$GETQUI("CANCEL_OPERATION")
$ LOOP1:
$  QNAME = F$GETQUI("DISPLAY_QUEUE","QUEUE_NAME","*","GENERIC,BATCH")
$  IF QNAME .EQS. "" THEN EXIT
$  WRITE SY$OUTPUT "Jobs in generic batch queue ", QNAME, " are:"
$ LOOP2:
$  JNAME = F$GETQUI("DISPLAY_JOB","JOB_NAME",,"ALL_JOBS")
$  IF JNAME .EQS. "" THEN goto LOOP1
$  WRITE SYS$OUTPUT "    ", JNAME
$ GOTO LOOP2
```

This sample command procedure searches through generic batch queues and displays all jobs currently residing in each queue. Because a wildcard queue name is specified ("*"), wildcard queue context is maintained across calls to F$GETQUI. This context is dissolved when the list of matching queues is exhausted. Furthermore, F$GETQUI returns a null string ("") to denote that no more objects match the specified search criteria. Finally, an initial cancel operation is performed to dissolve any wildcard context for the process that may still exist from a previously aborted search sequence (for example, abort of a SHOW QUEUE command or the running of this command procedure).

F$GETSYI

Invokes the $GETSYI system service to return status and identification information about the local system (or about a node in the local VAXcluster, if your system is part of a VAXcluster).

FORMAT

F$GETSYI*(item [,node])*

return value

Either an integer or a character string, depending on the item you request.

ARGUMENTS

item

Indicates the type of information to be reported about the local node (or about another node in your VAXcluster, if your system is part of a VAXcluster). Specify the item as a character string expression.

You can specify the items in Table DCL–11 only for your local node; you cannot specify the node argument with these items. You can specify these items whether or not you are in a VAXcluster.

You can specify the items in Table DCL–12 for either your local node or for another node in your VAXcluster. The information in this table is returned for your local node if you do not specify the node argument; the information is returned for the specified node if you include the node argument. Your system must be a member of a VAXcluster in order to specify the items in this table, except for CLUSTER_MEMBER. You can specify CLUSTER_MEMBER whether or not your system is a member of a VAX_CLUSTER.

You can also specify any of the SYSGEN parameters listed in Section A of the *VMS System Generation Utility Manual*. However, you can specify SYSGEN parameters only for your local node; you cannot specify the node argument with these items.

node

Specifies the node in your VAXcluster for which information is to be returned. Specify the node as a character string expression. (This argument can be specified only if your system is part of a VAXcluster.)

You can request information about another node in your VAXcluster only when you specify an item from Table DCL–12. If you do not specify a node, the default is the current node.

You cannot use wildcards to specify the node argument with the F$GETSYI function (as you can with the $GETSYI system service).

DESCRIPTION

The F$GETSYI returns information on the items that can be specified with the $GETSYI system service. See the *VMS System Services Reference Manual* for more information on the $GETSYI system service.

Table DCL–11 lists the items you can specify with the F$GETSYI lexical function to get information about your local node. Table DCL–12 lists the items you can specify to get information about either your local node, or another node in your VAXcluster.

Table DCL–11 F$GETSYI Items for the Local Node Only

Item	Return Type	Information Returned
ARCHFLAG	String	Architecture flags for the system.
BOOTTIME	String	The time the system was booted.
CHARACTER_EMULATED	String	"TRUE" or "FALSE" to indicate whether the character string instructions are emulated on the CPU.
CPU	Integer	The processor type, as represented in the processor's SID register. For example, the integer 1 represents a VAX–11/780 and the integer 6 represents a VAX 8530, VAX 8550, VAX 8700 and VAX 8800.
DECIMAL_EMULATED	String	"TRUE" or "FALSE" to indicate whether the decimal string instructions are emulated on the CPU.
D_FLOAT_EMULATED	String	"TRUE" or "FALSE" to indicate whether the D_floating instructions are emulated on the CPU.
ERRORLOGBUFFERS	Integer	Number of system pages in use as buffers for error logging.
F_FLOAT_EMULATED	String	"TRUE" or "FALSE" to indicate whether the F_floating instructions are emulated on the CPU.
G_FLOAT_EMULATED	String	"TRUE" or "FALSE" to indicate whether the G_floating instructions are emulated on the CPU.
PAGEFILE_FREE	Integer	Number of free pages in the currently installed paging files.
PAGEFILE_PAGE	Integer	Number of pages in the currently installed paging files.
SID	Integer	System identification register.
SWAPFILE_FREE	Integer	Number of free pages in the currently installed swapping files.
SWAPFILE_PAGE	Integer	Number of pages in the currently installed swapping files.
VERSION	String	Version of VMS in use (8-character string filled with trailing blanks).

Table DCL–12 F$GETSYI Items for the Local Node or for Other Nodes in the VAXCluster

Item	Return Type	Information Returned
ACTIVECPU_CNT	Integer	The count of CPUs actively participating in the current boot of the SMP system.
AVAILCPU_CNT	Integer	The count of CPUs recognized in the system.
CLUSTER_FSYSID	String	System identification number for first node to boot in the VAXcluster (the founding node). This number is returned as a character string containing a hexadecimal number.
CLUSTER_FTIME	String	The time when the first node in the VAXcluster was booted.
CLUSTER_ MEMBER	String	"TRUE" or "FALSE" if the node is a member of the local VAXcluster.
CLUSTER_NODES	Integer	Total number of nodes in the VAXcluster, as an integer.
CLUSTER_ QUORUM	Integer	Total quorum for the VAXcluster.
CLUSTER_VOTES	Integer	Total number of votes in the VAXcluster.
CONTIG_ GBLPAGES	Integer	Total number of free, contiguous global pages.
FREE_GBLPAGES	Integer	Current count of free global pages.
FREE_GBLSECTS	Integer	Current count of free global section table entries.
HW_MODEL	Integer	An integer that identifies the node's VAX model type.
HW_NAME	String	The VAX model name.
NODENAME	String	Node name.
NODE_AREA	Integer	The VAX DECnet area for the target node.
NODE_CSID	String	The CSID of the specified node, as a string containing a hexadecimal number. The CSID is a form of system identification.
NODE_HWTYPE	String	Hardware type of the specified node.
NODE_HWVERS	String	Hardware version of the specified node.
NODE_NUMBER	Integer	The VAX DECnet number for the specified node.
NODE_QUORUM	Integer	Quorum that the node has.
NODE_SWINCARN	String	Software incarnation number for the specified node. This number is returned as a string containing a hexadecimal number.
NODE_SWTYPE	String	Type of operating system software used by the specified node.

Table DCL–12 (Cont.) F$GETSYI Items for the Local Node or for Other Nodes in the VAXCluster

Item	Return Type	Information Returned
NODE_SWVERS	String	Software version of the specified node.
NODE_SYSTEMID	String	System identification number for the specified node. This number is returned as a string containing a hexadecimal number.
NODE_VOTES	Integer	Number of votes that the node has.
SCS_EXISTS	String	"TRUE" or "FALSE" to indicate whether the system communication subsystem (SCS) is currently loaded on a VAX node.

EXAMPLES

1
```
$ SYSID = F$GETSYI("SID")
$ SHOW SYMBOL SYSID
  SID = 19923201  Hex = 01300101 Octal = 000401
```

This example shows how to use the F$GETSYI function to return the information in the system identification register. Use quotation marks around the argument SID because it is a string literal. The value returned by the F$GETSYI function is assigned to the symbol SYSID. Because a node is not specified, information about your current node is returned.

2
```
$ MEM = F$GETSYI("CLUSTER_MEMBER", "LONDON")
$ SHOW SYMBOL MEM
  MEM = "TRUE"
```

This example uses the F$GETSYI function to determine whether the node LONDON is a member of the local VAXcluster. The "TRUE" indicates that the remote node LONDON is a member of the VAXcluster.

3
```
$ LIM = F$GETSYI("BJOBLIM")
$ SHOW SYMBOL LIM
  LIM = 16    Hex = 00000010  Octal = 00000000020
```

This example uses the SYSGEN parameter BJOBLIM as an argument for the F$GETSYI function. This argument returns the batch job limit for the current system.

F$IDENTIFIER

Converts an alphanumeric identifier to its integer equivalent, or converts an integer identifier to its alphanumeric equivalent. An identifier is a name or number that identifies a category of users of a data resource. The system uses identifiers to determine a user's access to a resource.

FORMAT

F$IDENTIFIER*(identifier,conversion-type)*

return value

An integer value if you are converting an identifier from a name to an integer. The F$IDENTIFIER function returns a string if you are converting an identifier from an integer to a name. If you specify an identifier that is not valid, the F$IDENTIFIER function returns a null string (if you are converting from number to name) or a zero (if you are converting from name to number).

ARGUMENTS

identifier

Specifies the identifier to be converted. Specify the identifier as an integer expression if you are converting an integer to a name. Specify the identifier as a character string expression if you are converting a name to an integer.

The F$IDENTIFIER function does not convert letters in the identifier to uppercase. Therefore, you must specify the identifier the same way it is defined in the "rights database". If the identifier is defined using uppercase letters, you must use uppercase letters when you specify the identifier for the F$IDENTIFIER function.

conversion-type

Indicates the type of conversion to be performed. If the identifier argument is alphanumeric, specify the translation argument as a character string containing "NAME_TO_NUMBER". If the identifier argument is numeric, specify the translation argument as a character string containing "NUMBER_TO_NAME".

EXAMPLES

1
```
$ UIC_INT= F$IDENTIFIER("SLOANE","NAME_TO_NUMBER")
$ SHOW SYMBOL UIC_INT
  UIC_INT = 15728665   Hex = 00F00019  Octal = 00074000031
$ UIC = F$FAO("!%U",UIC_INT)
$ SHOW SYMBOL UIC
  UIC = [360,031]
```

This example uses the F$IDENTIFIER to convert the member identifier from the UIC [MANAGERS,SLOANE] to an integer. The F$IDENTIFIER function shows that the member identifier SLOANE is equivalent to the integer 15728665. Note that you must specify the identifier SLOANE using uppercase letters.

To convert this octal number to a standard numeric UIC, use the F$FAO function with the !%U directive. (This directive converts a longword to a UIC in named format.) In this example, the member identifier SLOANE is equivalent to the numeric UIC [360,031].

2
```
$ UIC_INT = (%O31 + (%X10000 * %O360))
$ UIC_NAME = F$IDENTIFIER(UIC_INT,"NUMBER_TO_NAME")
$ SHOW SYMBOL UIC_NAME
  UIC_NAME = "ODONNELL"
```

This example obtains the alphanumeric identifier associated with the numeric UIC [360,031]. First, you must obtain the longword integer that corresponds to the UIC [360,031]. To do this, place the member number into the low order word. Place the group number into the high order word. Next, use the F$IDENTIFIER function to return the named identifier associated with the integer.

F$INTEGER

Returns the integer equivalent of the result of the specified expression.

FORMAT

F$INTEGER*(expression)*

return value

An integer value that is equivalent to the specified expression.

ARGUMENT

expression

Specifies the expression to be evaluated. Specify either an integer or a character string expression.

If you specify an integer expression, the F$INTEGER function evaluates the expression and returns the result. If you specify a string expression, the F$INTEGER function evaluates the expression, converts the resulting string to an integer, and returns the result.

After evaluating a string expression, the F$INTEGER function converts the result to an integer in the following way. If the resulting string contains characters that form a valid integer, the F$INTEGER function returns the integer value. If the string contains characters that do not form a valid integer, the F$INTEGER function returns the integer 1 if the string begins with T, t, Y, or y. The function returns the integer 0 if the string begins with any other character.

EXAMPLE

```
$ A = "23"
$ B = F$INTEGER("-9" + A)
$ SHOW SYMBOL B
  B = -923 Hex=FFFFFC65 Octal=176145
```

This example shows how to use the F$INTEGER function to equate a symbol to the integer value returned by the function.

The F$INTEGER function in the above example returns the integer equivalent of the string expression ("–9" + A). First, the F$INTEGER function evaluates the string expression by concatenating the string literal "–9" with the string literal "23". Note that the value of the symbol A is automatically substituted in a string expression. Also note that the plus sign (+) is a string concatenation operator since both arguments are string literals.

After the string expression is evaluated, the F$INTEGER function converts the resulting character string ("–923") to an integer, and returns the value –923. This integer value is assigned to the symbol B.

F$LENGTH

Returns the length of the specified character string.

FORMAT

F$LENGTH*(string)*

return value

An integer value for the length of the string.

ARGUMENT

string
Specifies the character string whose length is being determined. Specify the string argument as a character string expression.

EXAMPLE

```
$ MESSAGE = F$MESSAGE(%X1C)
$ SHOW SYMBOL MESSAGE
  MESSAGE = "%SYSTEM-F-EXQUOTA, exceeded quota"
$ STRING_LENGTH = F$LENGTH(MESSAGE)
$ SHOW SYMBOL STRING_LENGTH
  STRING_LENGTH = 33   Hex = 00000021  Octal = 000041
```

The first assignment statement uses the F$MESSAGE function to return the message that corresponds to the hexadecimal value 1C. The message is returned as a character string and is assigned to the symbol MESSAGE.

The F$LENGTH function is then used to return the length of the character string assigned to the symbol MESSAGE. You do not need to use quotation marks when you use the symbol MESSAGE as an argument for the F$LENGTH function. (Quotation marks are not used around symbols in character string expressions.)

The F$LENGTH function returns the length of the character string and assigns it to the symbol STRING_LENGTH. At the end of the example, the symbol STRING_LENGTH has a value equal to the number of characters in the value of the symbol named MESSAGE, that is, 33.

F$LOCATE

Locates a specified portion of a character string and returns as an integer the offset of the first character. If the substring is not found, F$LOCATE returns the length (the offset of the last character in the character string plus one) of the searched string.

FORMAT

F$LOCATE*(substring,string)*

return value

An integer value representing the offset of the substring argument. An offset is the position of a character or a substring relative to the beginning of the string. The first character in a string is always offset position 0 from the beginning of the string (which always begins at the leftmost character).

If the substring is not found, the F$LOCATE function returns an offset of the last character in the character string plus 1. (This equals the length of the string.)

ARGUMENTS

substring

The character string, specified in the string argument, that you want to locate within the string.

string

The character string to be edited by F$LOCATE.

EXAMPLES

1
```
$ FILE_SPEC = "MYFILE.DAT;1"
$ NAME_LENGTH = F$LOCATE(".",FILE_SPEC)
```

The F$LOCATE function in this example returns the position of the period in the string with respect to the beginning of the string. The period is in offset position 6, so the value 6 is assigned to the symbol NAME_LENGTH. Note that NAME_LENGTH also equals the length of the file name portion of the file specification MYFILE.DAT, that is, 6.

The substring argument, the period, is specified as a string literal and is therefore enclosed in quotation marks. The argument FILE_SPEC is a symbol, so it should not be placed within quotation marks. It is automatically replaced by its current value during the processing of the function.

```
2   $ INQUIRE TIME "Enter time"
    $ IF F$LOCATE(":",TIME) .EQ. F$LENGTH(TIME) THEN -
      GOTO NO_COLON
```

This section of a command procedure compares the results of the F$LOCATE and F$LENGTH functions to see if they are equal. This technique is commonly used to determine whether a character or substring is contained in a string.

In the example, the INQUIRE command prompts for a time value and assigns the user-supplied time to the symbol TIME. The IF command checks for the presence of a colon in the string entered in response to the prompt. If the value returned by the F$LOCATE function equals the value returned by the F$LENGTH function, the colon is not present. You use the .EQ. operator (rather than .EQS.) because the F$LOCATE and F$LENGTH functions return integer values.

Note that quotation marks are used around the substring argument, the colon, because it is a string literal. However, the symbol TIME does not require quotation marks because it is automatically evaluated as a string expression.

F$LOGICAL

Translates a logical name and returns the equivalence name string (or a null string if no match is found). The translation is not iterative; the equivalence string is not checked to determine whether it is a logical name.

As of VMS Version 4.0, the F$LOGICAL lexical function is superseded by F$TRNLNM. DIGITAL recommends using the F$TRNLNM lexical function. See F$TRNLNM for a complete description of the F$TRNLNM lexical function.

FORMAT

F$LOGICAL*(logical-name)*

F$MESSAGE

Returns as a character string the facility, severity, identification, and text associated with the specified system status code.

FORMAT

F$MESSAGE*(status-code)*

return value

A character string containing the system message that corresponds to the argument you specify.

Note that, although each message in the system message file has a numeric value or range of values associated with it, there are many possible numeric values that do not have corresponding messages. If you specify an argument that has no corresponding message, the F$MESSAGE function returns a string containing the NOMSG error message.

For more information on system error messages, see the *VMS System Messages and Recovery Procedures Reference Volume*.

ARGUMENT

status-code

The status code for which you are requesting error message text. You must specify the status code as an integer expression.

EXAMPLE

```
$ ERROR_TEXT = F$MESSAGE(%X1C)
$ SHOW SYMBOL ERROR_TEXT
  ERROR_TEXT = "%SYSTEM-F-EXQUOTA, exceeded quota"
```

This example shows how to use the F$MESSAGE function to determine the message associated with the status code %X1C. The F$MESSAGE function returns the message string, which is assigned to the symbol ERROR_TEXT.

F$MODE

Returns a character string showing the mode in which a process is executing. The F$MODE function has no arguments, but must be followed by parentheses.

FORMAT

F$MODE()

return value

The character string "INTERACTIVE" for interactive processes. If the process is noninteractive, the character string "BATCH", "NETWORK" or "OTHER" is returned. Note that the return string always contains uppercase letters.

ARGUMENTS

None.

DESCRIPTION

The F$MODE function is useful in command procedures that must operate differently when executed interactively and noninteractively. You should include either the F$MODE function or the F$ENVIRONMENT function in your login command file to execute different commands for interactive terminal sessions and noninteractive sessions.

If you do not include the F$MODE function to test whether your login command file is being executed from an interactive process, and the login command file is executed from a noninteractive process (such as a batch job), the process may terminate if the login command file contains commands that are appropriate only for interactive processing.

A command procedure can use the F$MODE function to test whether the procedure is being executed during an interactive terminal session. It can direct the flow of execution according to the results of this test.

EXAMPLE

```
$ IF F$MODE() .NES. "INTERACTIVE" THEN GOTO NON_INT_DEF
$ INTDEF:         ! Commands for interactive terminal sessions
    .
    .
    .
$ EXIT
$ NON_INT_DEF:          !Commands for non-interactive processes
    .
    .
    .
```

This example shows the beginning of a login.com file that has two sets of initialization commands: one for interactive mode and one for noninteractive mode (including batch and network jobs). The IF command compares the character string returned by F$MODE with the character string INTERACTIVE; if they are not equal, control branches to the label NON_INT_DEF. If the character strings are equal, the statements following the label INTDEF are executed and the procedure exits before the statements at NON_INT_DEF.

F$PARSE

Invokes the $PARSE RMS service to parse a file specification and return either the expanded file specification or the particular file specification field that you request.

FORMAT

F$PARSE*(file-spec [,default-spec] [,related-spec] [,field] [,parse-type])*

return value

A character string containing the expanded file specification or the field you specify. If you do not provide a complete file specification for the file-spec argument, the F$PARSE function supplies defaults in the return string, as described in the Description section.

If an error is detected during the parse, the F$PARSE function returns a null string, except when you specify a field name or the SYNTAX_ONLY parse type.

ARGUMENTS

file-spec

Specifies a character string containing the file specification to be parsed.

The file specification can contain wildcard characters. If you use a wildcard character, the file specification returned by the F$PARSE function contains the wildcard.

default-spec

Specifies a character string containing the default file specification.

The fields in the default file specification are substituted in the output string if a particular field in the file-spec argument is missing. You can make further substitutions in the file-spec argument by using the related-spec argument.

related-spec

Specifies a character string containing the related file specification.

The fields in the related file specification are substituted in the output string if a particular field is missing from both the file-spec and default-spec arguments.

field

Specifies a character string containing the name of a field in a file specification. Specifying the field argument causes F$PARSE to return a specific portion of a file specification.

Specify one of the following field names (do not abbreviate):

NODE	Node name
DEVICE	Device name
DIRECTORY	Directory name
NAME	File name
TYPE	File type
VERSION	File version number

parse-type

The type of parsing to be performed. By default, the F$PARSE function verifies that the directory in the file specification exists on the device in the file specification. Note that the device and directory can be explicitly given in one of the arguments, or can be provided by default.

Also, by default the F$PARSE function translates logical names if they are provided in any of the arguments. The F$PARSE function stops iterative translation when it encounters a logical name with the CONCEALED attribute.

You can change how the F$PARSE function parses a file specification by using one of the following keywords:

NO_CONCEAL	Ignores the "conceal" attribute in the translation of a logical name as part of the file specification; that is, logical name translation does not end when a concealed logical name is encountered.
SYNTAX_ONLY	The syntax of the file specification is checked without verifying that the specified directory exists on the specified device.

DESCRIPTION

The F$PARSE function parses file specifications using the $PARSE RMS service. For more information on the $PARSE routine, see the *VMS Record Management Services Manual*.

When you use the F$PARSE function, you can omit those optional arguments to the right of the last argument you specify. However, you must include commas as placeholders if you omit optional arguments to the left of the last argument you specify.

If you omit the device and directory names in the file-spec argument, the F$PARSE function supplies defaults, first from the default-spec argument and second from the related-spec argument. If names are not provided by these arguments, the F$PARSE function uses your current default disk and directory.

If you omit the file name, file type, or version number, the F$PARSE function supplies defaults, first from the default-spec argument and second from the related-spec argument. If names are not provided by these arguments, the F$PARSE function returns a null specification for these fields.

EXAMPLES

1

```
$ SET DEF DISK2:[FIRST]
$ SPEC = F$PARSE("JAMES.MAR","[ROOT]",,,"SYNTAX_ONLY")
$ SHOW SYMBOL SPEC
  SPEC = "DISK2:[ROOT]JAMES.MAR;"
```

In this example, the F$PARSE function returns the expanded file specification for the file JAMES.MAR. The example uses the SYNTAX_ONLY keyword to request that F$PARSE should check the syntax, but should not verify that the [ROOT] directory exists on DISK2.

The default device and directory are DISK2:[FIRST]. Because the directory name [ROOT] is specified as the default-spec argument in the assignment statement, it is used as the directory name in the output string. Note that the default device returned in the output string is DISK2, and the default version number for the file is null. You must place quotation marks around the arguments JAMES.MAR and ROOT because they are string literals.

If you had not specified syntax-only parsing, and [ROOT] were not on DISK2, a null string would have been returned.

2

```
$ SET DEFAULT DB1:[VARGO]
$ SPEC = F$PARSE("INFO.COM",,,"DIRECTORY")
$ SHOW SYMBOL SPEC
  SPEC = "[VARGO]"
```

In this example the F$PARSE function returns the directory name of the file INFO.COM. Note that because the default-spec and related-spec are omitted from the argument list, commas must be inserted in their place.

3

```
$ SPEC= F$PARSE("DENVER::DB1:[PROD]RUN.DAT",,,"TYPE")
$ SHOW SYMBOL SPEC
  SPEC = ".DAT"
```

In this example, the F$PARSE function is used to parse a file specification containing a node name. The F$PARSE function returns the file type DAT for the file RUN.DAT at the remote node DENVER.

F$PID

The F$PID function returns a process identification (PID) number and updates the context symbol to point to the current position in the system's process list. The PIDs returned by the F$PID function depend on the privilege of your process. If you have GROUP privilege, the F$PID function returns PIDs of processes in your group. If you have WORLD privilege, the F$PID function returns PIDs of all processes on the system. If you lack GROUP or WORLD privileges, the F$PID function returns only your process PID.

FORMAT

F$PID*(context-symbol)*

return value

A character string containing the process identification (PID) number of a process in the system's list of processes.

The PIDs returned by the F$PID function depend on the privilege of your process. If you have GROUP privilege, the F$PID function returns PIDs of processes in your group. If you have WORLD privilege, the F$PID function returns PIDs of all processes on the system. If you lack GROUP or WORLD privileges, the F$PID function returns only your process PID.

After the last PID in the system's process list is returned, the F$PID function returns a null string.

ARGUMENT

context-symbol

Specifies a symbol that DCL uses to store a pointer into the system's list of processes. The F$PID function uses this pointer to return a PID.

Specify the context-symbol by using a symbol. The first time you use the F$PID function in a command procedure, you should use a symbol that is either undefined or equated to the null string ("").

If the context-symbol is undefined or equated to a null string (""), the F$PID function returns the first PID in the system's process list that it has the privilege to access. That is, if you have GROUP privilege and if the context-symbol is null or undefined, the F$PID function returns the PID of the first process in your group. If you have WORLD privilege, the F$PID function returns the PID of the first process in the list. If you have neither GROUP nor WORLD privileges, the F$PID returns your process PID.

DESCRIPTION

Use the F$PID function to obtain the PIDs of all processes in your group (if you have GROUP privilege) or on the system (if you have WORLD privilege.)

The first time you use the F$PID function, use a symbol that is either undefined or equated to the null ("") string. This causes the F$PID function to return the first PID in the system's process list that you have the privilege to access. It also causes the F$PID function to initialize the context-symbol.

Once the context-symbol is initialized, each subsequent F$PID function returns the next PID in sequence, and updates the context symbol. After the last PID in the process list is returned, the F$PID function returns a null string.

EXAMPLE

```
$ CONTEXT = ""
$ START:
$     PID = F$PID(CONTEXT)
$     IF PID .EQS. "" THEN EXIT
$     SHOW SYMBOL PID
$     GOTO START
```

This command procedure uses the F$PID function to display a list of PIDs. The assignment statement declares the symbol CONTEXT, which is used as the context-symbol argument for the F$PID function. Because CONTEXT is equated to a null string, the F$PID function returns the first PID in the process list that it has the privilege to access.

The PIDs displayed by this command procedure depend on the privilege of your process. When run with GROUP privilege, the PIDs of users in your group are displayed. When run with WORLD privilege, the PIDS of all users on the system are displayed. Without GROUP or WORLD privilege, only your PID is displayed.

F$PRIVILEGE

Returns a value of either "TRUE" or "FALSE", depending on whether your current process privileges match those specified in the argument. You can specify either the positive or negative version of a privilege.

FORMAT

F$PRIVILEGE*(priv-states)*

return value

A character string containing the value "TRUE" or "FALSE". The F$PRIVILEGE function returns the string "FALSE" if any one of the privileges in the priv-states list is false.

ARGUMENT

priv-states

A character string containing a privilege or a list of privileges separated by commas. For a list of process privileges, see Table A-1 in the *VMS DCL Concepts Manual*. Specify any one of the process privileges except [NO]ALL.

DESCRIPTION

Use the F$PRIVILEGE function to identify your current process privileges.

If "NO" precedes the privilege, the privilege must be disabled in order for the function to return a value of "TRUE". The F$PRIVILEGE function checks each of the keywords in the specified list, and if the result for any one is false, the string "FALSE" is returned.

EXAMPLE

```
$ PROCPRIV = F$PRIVILEGE("OPER,GROUP,TMPMBX,NONETMBX")
$ SHOW SYMBOL PROCPRIV
  PROCPRIV = "FALSE"
```

The F$PRIVILEGE function is used to test whether the process has OPER, USER, TMPMBX, and NETMBX privileges.

The process in this example has OPER, GROUP, TMPMBX, and NETMBX privileges. Therefore, a value of "FALSE" is returned because the process has NETMBX privilege, but NONETMBX was specified in the priv-states list. Although the Boolean result for the other three keywords is true, the entire expression is declared false because the result for NONETMBX was false.

F$PROCESS

Obtains the current process name string. The F$PROCESS function has no arguments, but must be followed by parentheses.

FORMAT F$PROCESS()

return value A character string containing the current process name.

ARGUMENTS *None.*

EXAMPLE

```
$ NAME = F$PROCESS()
$ SHOW SYMBOL NAME
  NAME = "MARTIN"
```

In this example, the F$PROCESS function returns the current process name and assigns it to the symbol NAME.

F$SEARCH

Invokes the $SEARCH RMS service to search a directory file and return the full file specification for a file you specify.

FORMAT

F$SEARCH *(file-spec[,stream-id])*

return value

A character string containing the expanded file specification for the file-spec argument. If the F$SEARCH function does not find the file in the directory, the function returns a null ("") string.

ARGUMENTS

file-spec

Specifies a character string containing the file specification to be searched for. If the device or directory names are omitted, the defaults from your current default disk and directory are used. The F$SEARCH function does not supply defaults for a file name or type. If the version is omitted, the specification for the file with the highest version number is returned. If the file-spec argument contains wildcards, each time F$SEARCH is called, the next file specification that agrees with the file-spec argument is returned. A null string is returned after the last file specification that agrees with the file-spec argument.

stream-id

A positive integer representing the search stream identification number.

The search stream identification number is used to maintain separate search contexts when you use the F$SEARCH function more than once and when you supply different file-spec arguments. If you use the F$SEARCH function more than once in a command procedure and if you also use different file-spec arguments, specify stream-id arguments to identify each search separately.

If you omit *stream-id*, the F$SEARCH function assumes an implicit single search stream. That is, the F$SEARCH function starts searching at the beginning of the directory file each time you specify a different file-spec argument.

DESCRIPTION

The F$SEARCH function allows you to search for files in a directory using the $SEARCH RMS service. For more information on the $SEARCH routine, see the *VMS Record Management Services Manual*.

You can use the F$SEARCH function in a loop in a command procedure to return file specifications for all files that match a file-spec argument containing a wildcard. Each time the F$SEARCH function is executed, it returns the next file specification that matches the wildcarded file specification. After the last file specification is returned, the next F$SEARCH function returns a null string. When you use the F$SEARCH function in a loop, you must include a wildcard in the file-spec argument. Otherwise, the F$SEARCH always returns the same file specification.

Note that you must maintain the context of the search stream explicitly (by stating a stream-id) or implicitly (by omitting the stream-id and using the same file-spec argument each time you execute the F$SEARCH function).

EXAMPLES

1

```
$ START:
$     FILE = F$SEARCH("SYS$SYSTEM:*.EXE")
$     IF FILE .EQS. "" THEN EXIT
$     SHOW SYMBOL FILE
$     GOTO START
```

This command procedure displays the file-specs of the latest version of all .EXE files in the SYS$SYSTEM directory. (Only the latest version is returned because a wildcard is not used as the version number.) The file-spec argument SYS$SYSTEM:*.EXE is surrounded by quotation marks because it is a character string expression.

Because no stream-id argument is specified, the F$SEARCH uses a single search stream. Each subsequent F$SEARCH function uses the same file-spec argument to return the next file specification of an .EXE file from SYS$SYSTEM:. After the latest version of each .EXE file has been displayed, the F$SEARCH function returns a null string and the procedure exits.

2

```
$ START:
$    COM = F$SEARCH ("*.COM;*",1)
$    DAT = F$SEARCH ("*.DAT;*",2)
$    SHOW SYMBOL COM
$    SHOW SYMBOL DAT
$    IF (COM.EQS. "") .AND. (DAT.EQS. "") THEN EXIT
$    GOTO START
```

This command procedure searches the default disk and directory for both COM and DAT files. Note that the stream-id is specified for each F$SEARCH function so that the context for each search is maintained.

The first F$SEARCH function starts searching from the top of the directory file for a file with a type of COM. When it finds a COM file, a pointer is set to maintain the search context. When the F$SEARCH function is used the second time, it again starts searching from the top of the directory file for a file with a type of DAT. When the procedure loops back to the label START, the stream-id argument allows each F$SEARCH function to start searching in the correct place in the directory file. After all versions of COM and DAT files are returned, the procedure exits.

3

```
$ FILESPEC = F$SEARCH("TRNTO""SMITH SALLY""::DBA1:[PROD]*.DAT")
$ SHOW SYMBOL FILESPEC
  FILESPEC = "TRNTO"smith password"::DBA1:[PROD]CARS.DAT"
```

This example uses the F$SEARCH function to return a file specification for a file at a remote node. The access control string is enclosed in double quotation marks because it is part of a character string expression when it is an argument for the F$SEARCH function. To include quotation marks in a character string expression, you must use two sets of quotation marks.

Note that, when the F$SEARCH function returns a node name containing an access control string, it substitutes the word "password" for the actual user password.

F$SETPRV

Invokes the $SETPRV system service to enable or disable specified user privileges. The F$SETPRV function returns a list of keywords indicating user privileges; this list shows the status of the specified privileges before F$SETPRV was executed.

Your process must be authorized to set the specified privilege. For detailed information on privilege restrictions, see the description of the $SETPRV system service in the *VMS System Services Reference Manual.*

FORMAT

F$SETPRV*(priv-states)*

return value

A character string containing keywords for the current process privileges before they were changed by the F$SETPRV function.

ARGUMENT

priv-states

A character string defining a privilege or a list of privileges separated by commas.

For a list of process privileges, see Table A-1 in the *VMS DCL Concepts Manual.*

DESCRIPTION

The F$SETPRV function returns keywords for your current privileges, whether or not you are authorized to change the privileges listed in the priv-states argument. However, the F$SETPRV function enables or disables only the privileges you are authorized to change.

When you run programs or execute procedures that include the F$SETPRV function, be sure that F$SETPRV restores your process to its proper privileged state. For additional information, refer to the examples that follow.

EXAMPLES

1
```
$ OLDPRIV = F$SETPRV("OPER,NOTMPMBX")
$ SHOW SYMBOL OLDPRIV
  OLDPRIV = "NOOPER,TMPMBX"
```

In this example, the process is authorized to change the OPER and TMPMBX privileges. The F$SETPRV function enables the OPER privilege and disables the TMPMBX privilege. In addition, the F$SETPRV function returns the keywords NOOPER and TMPMBX, showing the state of these privileges before they were changed.

You must place quotation marks around the list of privilege keywords because it is a string literal.

2
```
$ SHOW PROCESS/PRIVILEGE

23-OCT-1988 15:55:09.60   RTA1:              User: JACKSON

Process privileges:

Process rights identifiers:
 INTERACTIVE
 LOCAL

$ NEWPRIVS = F$SETPRV("ALL, NOOPER")
$ SHOW SYMBOL NEWPRIVS
  NEWPRIVS = "NOCMKRNL,NOCMEXEC,NOSYSNAM,NOGRPNAM,NOALLSPOOL,NODETACH,
      NODIAGNOSE,NOLOG_IO,NOGROUP,NOACNT,NOPRMCEB,NOPRMMBX,NOPSWAPM,
      NOALTPRI,NOSETPRV,NOTMPMBX,NOWORLD,NOMOUNT,NOOPER,NOEXQUOTA,
      NONETMBX,NOVOLPRO,NOPHY_IO,NOBUGCHK,NOPRMGBL,NOSYSGBL,NOPFNMAP,
      NOSHMEM,NOSYSPRV,NOBYPASS,NOSYSLCK,NOSHARE,NOUPGRADE,NODOWNGRADE,
      NOGRPPRV,NOREADALL,NOSECURITY,OPER"
$ SHOW PROCESS/PRIVILEGE

23-OCT-1988 15:59:19.30   RTA1:              User: JACKSON

Process privileges:
 CMKRNL               may change mode to kernel
 CMEXEC               may change mode to exec
 SYSNAM               may insert in system logical name table
 GRPNAM               may insert in group logical name table
 ALLSPOOL             may allocate spooled device
 DETACH               may create detached processes
 DIAGNOSE             may diagnose devices
 LOG_IO               may do logical i/o
 GROUP                may affect other processes in same group
 ACNT                 may suppress accounting message
 PRMCEB               may create permanent common event clusters
 PRMMBX               may create permanent mailbox
 PSWAPM               may change process swap mode
 ALTPRI               may set any priority value
 SETPRV               may set any privilege bit
 TMPMBX               may create temporary mailbox
 WORLD                may affect other processes in the world
 MOUNT                may execute mount acp function
 EXQUOTA              may exceed quota
 NETMBX               may create network device
 VOLPRO               may override volume protection
 PHY_IO               may do physical i/o
 BUGCHK               may make bug check log entries
 PRMGBL               may create permanent global sections
 SYSGBL               may create system wide global sections
 PFNMAP               may map to specific physical pages
 SHMEM                may create/delete objects in shared memory
 SYSPRV               may access objects via system protection
 BYPASS               bypasses UIC checking
 SYSLCK               may lock system wide resources
 SHARE                may assign channels to non-shared device
 GRPPRV               group access via system protection
 READALL              may read anything as the owner
 SECURITY             may perform security functions

Process rights identifiers:
 INTERACTIVE
 LOCAL

$ NEWPRIVS = F$SETPRV(NEWPRIVS)
$ SHOW PROCESS/PRIVILEGE

23-OCT-1988 16:05:07.23   RTA1:              User: JACKSON
```

```
Process privileges:
 OPER                 operator privilege

Process rights identifiers:
 INTERACTIVE
 LOCAL
```

This example illustrates the methodology of the F$SETPRV lexical function.

First, the DCL command SHOW PROCESS/PRIVILEGE is used to determine the current process privileges. Note that the process has no privileges enabled.

The F$SETPRV function is then used to process the ALL keyword and enable all privileges recording the previous state of each privilege in the symbol NEWPRIVS. Next, F$SETPRV processes the NOOPER keyword and disables the OPER privilege, recording the previous state of OPER in NEWPRIVS. Note that the OPER privilege appears in the returned string twice: first as NOOPER and then as OPER.

Typing the command SHOW PROCESS/PRIVILEGE now shows that the current process has all privileges enabled except OPER.

If the returned string is used as the parameter to F$SETPRV, the process has the OPER privilege enabled. This occurs because the OPER command was present twice in the symbol NEWPRIVS. As a result, F$SETPRV looked at the first keyword NOOPER and disabled the privilege. Finally, after processing several other keywords in the NEWPRIVS string, the OPER keyword is presented allowing F$SETPRV to enable the OPER privilege.

If you are using the ALL or NOALL keywords to save your current privilege environment, DIGITAL recommends that you perform the following procedure to modify the process for a command procedure:

```
$ CURRENT_PRIVS = F$SETPRV("ALL")
$ TEMP = F$SETPRV("NOOPER")
```

If you use this procedure, you can then specify the following command statement at the end of your command procedure so that the original privilege environment is restored:

```
$ TEMP = F$SETPRV(CURRENT_PRIVS)
```

```
3  $ SAVPRIV = F$SETPRV("NOGROUP")
   $ SHOW SYMBOL SAVPRIV
     SAVPRIV = "GROUP"
   $ TEST = F$PRIVILEGE("GROUP")
   $ SHOW SYMBOL TEST
     TEST = "TRUE"
```

In this example, the process is not authorized to change the GROUP privilege. However, the F$SETPRV function still returns the current setting for the GROUP privilege.

The F$PRIVILEGE function is used to see whether the process has GROUP privilege. The return string, TRUE, indicates that the process has GROUP privilege, even though the F$SETPRV function attempted to disable the privilege.

F$STRING

Returns the string that is equivalent to the specified expression.

FORMAT

F$STRING*(expression)*

return value

A character string equivalent to the specified expression.

ARGUMENT

expression

The integer or string expression to be evaluated.

If you specify an integer expression, the F$STRING expression evaluates the expression, converts the resulting integer to a string, and returns the result. If you specify a string expression, the F$STRING expression evaluates the expression and returns the result.

When converting an integer to a string, the F$STRING function uses decimal representation and omits leading zeroes. When converting a negative integer, the F$STRING function places a minus sign at the beginning string representation of the integer.

EXAMPLE

```
$ A = 5
$ B = F$STRING(-2 + A)
$ SHOW SYMBOL B
  B = "3"
```

The F$STRING function in this example converts the result of the integer expression (–2 + A) to the numeric string, "3". First, the F$STRING function evaluates the expression (–2 + A). Note that 5, the value of symbol A, is automatically substituted when the integer expression is evaluated.

After the integer expression is evaluated, the F$STRING function converts the resulting integer, 3, to the string "3". This string is assigned to the symbol B.

F$TIME

Returns the current date and time in absolute time format.

The F$TIME function has no arguments, but must be followed by parentheses.

FORMAT F$TIME()

return value

A character string containing the current date and time. The returned string has the following fixed, 23-character format:

dd-mmm-yyyy hh:mm:ss.cc

When the current day of the month is any of the values 1 through 9, the first character in the returned string is a blank character. The time portion of the string is always in character position 13, at an offset of 12 characters from the beginning of the string.

Note that you must use the assignment operator (=) to preserve the blank character in the returned string. If you use the string assignment operator (:=), the leading blank is dropped.

ARGUMENTS

None.

EXAMPLE

```
$ OPEN/WRITE OUTFILE DATA.DAT
$ TIME_STAMP = F$TIME()
$ WRITE OUTFILE TIME_STAMP
```

This example shows how to use the F$TIME function to time-stamp a file that you create from a command procedure. OUTFILE is the logical name for the file DATA.DAT, which is opened for writing. The F$TIME function returns the current date and time string, and assigns this string to the symbol TIME_STAMP. The WRITE command writes the date and time string to OUTFILE.

F$TRNLNM

Translates a logical name and returns the equivalence name string, or the requested attributes of the logical name specified. The translation is not iterative; the equivalence string is not checked to determine whether it is a logical name.

FORMAT

F$TRNLNM*(logical-name [,table] [,index] [,mode] [,case] [,item])*

return value

The equivalence name or attribute of the specified logical name. The return value can be a character string or an integer, depending on the arguments you specify with the F$TRNLNM function. If no match is found, a null string is returned.

ARGUMENTS

logical-name

Specifies a character string containing the logical name to be translated.

table

Specifies a character string containing the logical name table or tables that the F$TRNLNM function should search to translate the logical name. The table argument must be a logical name that translates to a logical name table or to a list of table names.

If you do not specify a table, the default value is LNM$DCL_LOGICAL. That is, the F$TRNLNM function searches the tables whose names are equated to the logical name LNM$DCL_LOGICAL. Unless LNM$DCL_LOGICAL has been redefined for your process, the F$TRNLNM function searches the process, job, group, and system logical name tables, in that order, and returns the equivalence name for the first match found.

index

Specifies the number of the equivalence name to be returned if the logical name has more than one translation. The index refers to the equivalence strings in the order the names were listed when the logical name was defined.

The index begins with 0; that is, the first name in a list of equivalence names is referenced by the index 0.

If you do not specify the index argument, the default is 0.

mode

Specifies a character string containing one of the following access modes for the translation: USER (default), SUPERVISOR, EXECUTIVE, or KERNEL.

The F$TRNLNM function starts by searching for a logical name created with the access mode specified in the mode argument. If it does not find a match, the F$TRNLNM function searches for the name created with each inner access mode and returns the first match found. For example, two logical names can have the same name, but one name can be created with user access mode and the other name with executive access mode. If the mode argument is USER,

the F$TRNLNM function returns the equivalence string for the user mode, not the executive mode, logical name.

case

Specifies the type of case translation to be performed. Specify the case argument as either of the following character strings: CASE_BLIND (default) or CASE_SENSITIVE.

If the translation is case blind, the F$TRNLNM function first searches for a logical name with characters of the same case as the name argument. If no match is found, the F$TRNLNM function searches for an uppercase version of the name argument and the logical names it is searching. The result of the first successful translation is returned.

If the translation is case sensitive, the F$TRNLNM function searches only for a logical name with characters of the same case as the name argument. The F$TRNLNM function returns a null string if no exact match is found.

item

A character string containing the type of information that F$TRNLNM should return about the specified logical name. Specify one of the following items:

Item	Return Type	Information Returned
ACCESS_ MODE	String	One of the following access modes associated with the logical name: "USER", "SUPERVISOR", "EXECUTIVE", "KERNEL".
CONCEALED	String	Either "TRUE" or "FALSE" to indicate whether the CONCEALED attribute was specified with the /TRANSLATION_ATTRIBUTES qualifier when the logical name was created. The CONCEALED attribute is used to create a concealed logical name.
CONFINE	String	Either "TRUE" or "FALSE" to indicate whether the logical name is confined. If the logical name is confined (TRUE), then the name is not copied to subprocesses. If the logical name is not confined (FALSE), then the name is copied to subprocesses.
CRELOG	String	"TRUE" or "FALSE" to indicate whether the logical name was created with the $CRELOG system service or with the $CRELNM system service, using the CRELOG attribute. If the logical name was created with the $CRELOG system service or with the $CRELNM system service, using the CRELOG attribute, then the string "TRUE" is returned. Otherwise, the string "FALSE" is returned.

Item	Return Type	Information Returned
LENGTH	Integer	Length of the equivalence name associated with the specified logical name. If the logical name has more than one equivalence name, the F$TRNLNM function returns the length of the name specified by the index argument.
MAX_INDEX	Integer	The largest index defined for the logical name. The index shows how many equivalence names are associated with a logical name. The index is zero based; that is, the index 0 refers to the first name in a list of equivalence names.
NO_ALIAS	String	Either "TRUE" or "FALSE" to indicate whether the logical name has the NO_ALIAS attribute. The NO_ALIAS attribute means that a logical name must be unique within outer access mode.
TABLE	String	Either "TRUE" or "FALSE" to indicate whether the logical name is the name of a logical name table.
TABLE_NAME	String	Name of the table where the logical name was found.
TERMINAL	String	Either "TRUE" or "FALSE" to indicate whether the TERMINAL attribute was specified with the /TRANSLATION_ATTRIBUTES qualifier when the logical name was created. The TERMINAL attribute indicates that the logical name is not a candidate for iterative translation.
VALUE	String	Default. The equivalence name associated with the specified logical name. If the logical name has more than one equivalence name, the F$TRNLNM function returns the name specified by the index argument.

DESCRIPTION

The F$TRNLNM function offers the capability of the $TRNLNM system service. When you use the F$TRNLNM function, you can omit optional arguments that can be used to the right of the last argument you specify. However, you must include commas as placeholders if you omit optional arguments to the left of the last argument that you specify.

You can use the F$TRNLNM function in command procedures to save the current equivalence of a logical name and later restore it. You can also use it to test whether logical names have been assigned.

EXAMPLES

1

```
$ SAVE_DIR = F$TRNLNM("SYS$DISK")+F$DIRECTORY()
    .
    .
    .
$ SET DEFAULT 'SAVE_DIR'
```

The assignment statement concatenates the values returned by the F$DIRECTORY and F$TRNLNM functions, and assigns the resulting string to the symbol SAVE_DIR. The symbol SAVE_DIR consists of a full device and directory name string.

The argument SYS$DISK is enclosed in quotation marks because it is a character string. (The command interpreter treats all arguments that begin with alphabetic characters as symbols or lexical functions, unless the arguments are enclosed within quotation marks.) None of the optional arguments is specified, so the F$TRNLNM function uses the defaults.

At the end of the command procedure, the original default directory is reset. When you reset the directory, you must place apostrophes around the symbol SAVE_DIR to force symbol substitution.

2

```
$ DEFINE/TABLE=LNM$GROUP TERMINAL 'F$TRNLNM("SYS$OUTPUT")'
```

This example shows a line from a command procedure that (1) uses the F$TRNLNM function to determine the name of the current output device and (2) creates a group logical name table entry based on the equivalence string.

You must enclose the argument SYS$OUTPUT in quotation marks because it is a character string.

Also, in this example you must enclose the F$TRNLNM function in single quotes to force the lexical function to be evaluated. Otherwise, the DEFINE command does not automatically evaluate the lexical function.

3

```
$ RESULT = F$TRNLNM("INFILE","LNM$PROCESS",0,"SUPERVISOR",,"NO_ALIAS")
$ SHOW SYMBOL RESULT
  RESULT = "FALSE"
```

In this example, the F$TRNLNM function searches the process logical name table for the logical name INFILE. The function starts the search by looking for the logical name INFILE created in supervisor mode. If no match is found, the function looks for INFILE created in executive mode.

When a match is found, the F$TRNLNM function determines whether the name INFILE was created with the NO_ALIAS attribute. In this case, the "NO_ALIAS" attribute is not specified.

F$TYPE

Returns the data type of a symbol.

FORMAT

F$TYPE*(symbol-name)*

return value

The string "INTEGER" if the symbol is equated to an integer, or if the symbol is equated to a string whose characters form a valid integer. If the symbol is equated to a character string whose characters do not form a valid integer, the F$TYPE function returns the string "STRING". If the symbol is undefined, a null string is returned.

ARGUMENT

symbol
Specifies a character string or integer that references the name of the symbol to be evaluated.

EXAMPLES

1
```
$ NUM = "52"
$ TYPE = F$TYPE(NUM)
$ SHOW SYMBOL TYPE
  TYPE = "INTEGER"
```

This example uses the F$TYPE function to determine the data type of the symbol NUM. NUM is equated to the character string "52". Because the characters in the string form a valid integer, the F$TYPE function returns the string "INTEGER".

2
```
$ NUM = 52
$ TYPE = F$TYPE(NUM)
$ SHOW SYMBOL TYPE
  TYPE = "INTEGER"
```

In this example, the symbol NUM is equated to the integer 52. The F$TYPE function shows that the symbol has an integer data type.

3
```
$ CHAR = "FIVE"
$ TYPE = F$TYPE(CHAR)
$ SHOW SYMBOL TYPE
  TYPE = "STRING"
```

In this example, the symbol CHAR is equated to the character string "FIVE". Because the characters in this string do not form a valid integer, the F$TYPE function shows that the symbol has a string value.

F$USER

Returns the current user identification code (UIC) in named format as a character string. The F$USER function has no arguments, but must be followed by parentheses.

FORMAT F$USER()

return value

A character string containing the current user identification (UIC), including square brackets. The UIC is returned in the format [group-identifier, member-identifier].

ARGUMENTS

None.

EXAMPLE

```
$ UIC = F$USER()
$ SHOW SYMBOL UIC
  UIC = "[GROUP6,JENNIFER]"
```

In this example the F$USER function returns the current user identification code and assigns it to the symbol UIC.

F$VERIFY

Returns an integer value indicating whether the procedure verification setting is currently on or off. If used with arguments, the F$VERIFY function can turn the procedure and image verification settings on or off. You must include the parentheses after the F$VERIFY function whether or not you specify arguments.

FORMAT

F$VERIFY*([procedure-value] [,image-value])*

return value

The integer 0 if the procedure verification setting is off, or the integer 1 if the procedure verification setting is on.

ARGUMENTS

procedure-value

An integer expression with a value of 1 to turn procedure verification on, or 0 to turn procedure verification off.

When procedure verification is on, each DCL command line in the command procedure is displayed on the output device. Procedure verification allows you to verify that each command is executing correctly.

If you use the procedure-value argument, the function first returns the current procedure verification setting. Then the command interpreter turns the procedure verification on or off, as specified by the argument.

image-value

An integer expression with a value of 1 to turn image verification on, or 0 to turn image verification off.

When image verification is on, data lines in the command procedure are displayed on the output device.

DESCRIPTION

Using the F$VERIFY function in command procedures allows you to test the current procedure verification setting. For example, a command procedure can save the current procedure verification setting before changing it and then later restore the setting. In addition, you can construct a procedure that does not display (or print) commands, regardless of what the initial state of verification is.

When you use the F$VERIFY function, you can specify zero, one, or two arguments. If you do not specify any arguments, neither of the verification settings is changed. If you specify only the procedure-value argument, both procedure and image verification are turned on (if the value is 1) or off (if the value is 0).

If you specify both arguments, procedure and image verification are turned on or off independently. If you specify the image-value argument alone, only image verification is turned on or off. If you specify the image-value argument alone, you must precede the argument with a comma.

You can also use the F$ENVIRONMENT function with VERIFY_PROCEDURE or VERIFY_IMAGE as the argument. With the F$ENVIRONMENT function, you can determine either the procedure or image verification setting; the F$VERIFY function determines only the procedure verification setting.

EXAMPLES

1

```
$ SAVE_PROC_VERIFY = F$ENVIRONMENT("VERIFY_PROCEDURE")
$ SAVE_IMAGE_VERIFY = F$ENVIRONMENT("VERIFY_IMAGE")
$ SET NOVERIFY
   .
   .
   .
$ TEMP = F$VERIFY(SAVE_PROC_VERIFY, SAVE_IMAGE_VERIFY)
```

This example shows an excerpt from a command procedure. The first assignment statement assigns the current procedure verification setting to the symbol SAVE_PROC_VERIFY. The second assignment statement assigns the current image verification setting to the symbol SAVE_IMAGE_VERIFY.

Then, the SET NOVERIFY command disables procedure and image verification. Later, the F$VERIFY function resets the verification settings, using the original values (equated to the symbols SAVE_PROC_VERIFY and SAVE_IMAGE_VERIFY). The symbol TEMP contains the procedure verification before it is changed with the F$VERIFY function. (In this example the value of TEMP is not used.)

2

```
$ VERIFY = F$VERIFY(0)
   .
   .
   .
$ IF VERIFY .EQ. 1 THEN SET VERIFY
```

This example shows an excerpt from a command procedure that uses the F$VERIFY function to save the current procedure verification setting and to turn both procedure and image verification off. At the end of the command procedure, if procedure verification was originally on, both the procedure and image verification are turned on.

LIBRARY

Invokes the Librarian Utility to create, modify, or describe an object, macro, help, text, or shareable image library. For a complete description of the Librarian Utility, including information about the LIBRARY command and its qualifiers, see the *VMS Librarian Utility Manual*.

FORMAT

LIBRARY *library-file-spec [input-file-spec[,...]]*

LINK

Invokes the VMS Linker to link one or more object modules into a program image and defines execution characteristics of the image. For a complete description of the linker, including more information about the LINK command, see the *VMS Linker Utility Manual*.

FORMAT

LINK *file-spec[,...]*

PARAMETER

file-spec[,...]

Specifies one or more input files (wildcard characters not allowed). Input files may be object modules, libraries to be searched for external references or from which specific modules are to be included, shareable images to be included in the output image or option files to be read by the linker. Separate multiple input file specifications with commas (,) or plus signs (+). In either case, the linker creates a single image file.

If you omit the file type in an input file specification, the linker supplies default file types, based on the nature of the file. For object modules, file type OBJ is assumed.

DESCRIPTION

Before a source-language program can run on the VMS operating system, it must be translated into object code and then linked. The VMS Linker binds the object modules, together with any other necessary information, into an executable image.

To invoke the VMS Linker from DCL level, enter the LINK command and the command parameter.

For an executable image, the command parameter specifies one or more input files including object modules to be linked, libraries to be searched for external references or from which specific modules are to be included, and option files to be read by the linker. Note that you cannot specify a shareable image input file from the command line; it can only be specified from a statement within an options file that you name on the command line.

If you name several input files on the command line, separate the file specifications with commas, or plus signs.

For a shareable image, the command parameter specifies the name of the shareable image being created.

You can direct output to different types of files using appropriate command qualifiers, such as /EXECUTABLE, /SHAREABLE, /MAP, and /SYMBOL_TABLE. By default, linker output and messages are directed to the SYS$OUTPUT device. Unless you specify otherwise, output files take the name of the first input file in the command parameter.

In addition to the LINK command and the command parameter, the command line may include qualifiers that either modify the command itself, or further qualify the command parameter.

A command qualifier modifies the command itself and may be located anywhere on the command line. Its position does not alter its function.

Positional qualifiers indicate to the linker which of the files in the command parameter list are to be the object of the specified action. If you position the qualifier next to the command, all listed files are affected. To affect one or more files selectively, position the qualifier immediately after the appropriate file specification(s).

If you specify incompatible qualifiers, the linker either ignores the command and displays an error message, or it may ignore the incompatibility and permit the linking operation to continue.

QUALIFIERS

/BRIEF

Requests the linker to produce a brief map (memory allocation) file; valid only with the /MAP qualifier.

A brief form of the map contains the following information:

- A summary of the image characteristics
- A list of all object modules included in the image
- A summary of link-time performance statistics

/CONTIGUOUS
/NOCONTIGUOUS (default)

Controls whether the output image file is contiguous.

/CROSS_REFERENCE
/NOCROSS_REFERENCE (default)

Controls whether the memory allocation listing (map) contains a symbol cross-reference list with entries for each global symbol referenced in the image, its value, and all modules in the image that refer to it.

/DEBUG[=file-spec]
/NODEBUG

Controls whether a debugger is included in the output image.

If the object module contains local symbol table or traceback information, you can specify /DEBUG to include the information in the image as well. If the object module does not contain local symbol table or traceback information, only global symbols are available for symbolic debugging.

By default, VAX Symbolic Debugger is linked with the image. However, you may use the file-spec option to specify an alternate debugger (wildcard characters not allowed).

For information on using the VAX Symbolic Debugger, see the *VMS Debugger Manual*.

/EXECUTABLE[=file-spec]
/NOEXECUTABLE

Permits you to specify whether or not the linker creates an executable image.

By default the linker creates an executable image with the same file name as the first input file and a file type of EXE but this qualifier gives you the option of assigning the image a file specification (wildcard characters not allowed).

The placement of the command qualifier determines the output file specification defaults.

You can use /NOEXECUTABLE or /EXECUTABLE=NL: to test a set of qualifiers, options, or input object modules, without creating an image file. However, it is recommended that you use /EXECUTABLE=NL: because the linker will not process certain qualifiers if /NOEXECUTABLE is used.

/FULL

Requests the linker to produce a full map (memory allocation) listing; valid only with /MAP qualifier.

A full listing contains the following information:

- All the information included in the brief listing (see /BRIEF)
- Detailed descriptions of each program section and image section in the image file
- Lists of global symbols by name and by value

/HEADER

Provides a system image header when used with the /SYSTEM qualifier. All other images always have headers. However, by default, system images do not have headers.

/INCLUDE=(module-name[,...])

Positional qualifier.

Selects modules from the associated object module library or image library as input to the linking operation. No wildcard characters are allowed in the module name specifications.

At least one module name must be specified. If you specify several modules, separate them with commas and enclose the list in parentheses.

If you specify /INCLUDE, you can also specify /LIBRARY; the library is then searched for unresolved references.

/LIBRARY

Positional qualifier.

Indicates that the associated input file is a library (default file type OLB) whose modules should be searched to resolve undefined symbols. You are not permitted to specify a library as the first input file unless you also specify the /INCLUDE qualifier to indicate which modules in the library are to be included in the input. You can use both /INCLUDE and /LIBRARY to qualify a file specification. In this case, the explicit inclusion of modules occurs first, then the library is used to search for unresolved references.

/MAP[=file-spec]
/NOMAP

Permits you to specify whether or not a memory allocation listing (map) is produced and gives you the option of assigning it a file specification. In interactive mode, the default is /NOMAP; in batch mode, the default is /MAP.

You can specify the map's contents using either the /BRIEF, /FULL, or /CROSS_REFERENCE qualifiers. If you do not specify any of these qualifiers, the map contains the following information:

- All the information contained in a brief listing (see /BRIEF)
- A list of user-defined global symbols by name
- A list of user-defined program sections

When you specify /MAP, you can control the defaults applied to the output file specification, as described in the *VMS DCL Concepts Manual.*

/OPTIONS

Positional qualifier.

Indicates that the associated input file (default file type OPT) contains a list of linking options.

For complete details on the contents of an options file, see the *VMS Linker Utility Manual.*

/P0IMAGE

Directs the linker to create an image that is stored only in P0 address space together with the stack and the VMS RMS buffers that usually go in P1 address space. The /P0IMAGE qualifier is used to create executable images that modify P1 address space. See the *VAX Architecture Handbook* for a description of P0 and P1 address space.

/PROTECT

When used with the /SHAREABLE qualifier, the /PROTECT qualifier directs the linker to create a protected shareable image that can execute privileged change mode instructions even when it is linked into a nonprivileged executable image.

/SELECTIVE_SEARCH

Positional qualifier.

Use this qualifier when you want the linker to omit from the output image symbol table, all symbols from the associated input object module that are not needed to resolve outstanding references. These symbols are also excluded from the symbol table file, if /SYMBOL_TABLE is specified. The binary code in the object module is always included.

/SHAREABLE[=file-spec]
/NOSHAREABLE

Command qualifier.

By default, the linker creates an executable image. If you specify the /SHAREABLE qualifier, the linker creates a shareable image file instead. Optionally, you may designate a name for the output file; however, wildcard characters are not permitted.

Shareable images are not executable; however, they can be linked with object modules to create executable images. If you specify both the /EXECUTABLE and the /SHAREABLE qualifier, the /SHAREABLE qualifier takes precedence.

When you specify /SHAREABLE, you can control the defaults applied to the output file specification by the placement of the qualifier in the command.

To specify an input shareable image, the /SHAREABLE qualifier must be used as an input file qualifier in an options file. See the description of the linker in the *VMS Linker Utility Manual*.

/SHAREABLE
/SHAREABLE=NOCOPY

Positional qualifier.

Use this positional qualifier in the context of an options file only to identify an input file as a shareable image file. The keyword NOCOPY tells the linker not to bind a private copy of the shareable image to the executable image. /SHAREABLE and /SHAREABLE=NOCOPY are equivalent.

/SYMBOL_TABLE[=file-spec]
/NOSYMBOL_TABLE

The default is /NOSYMBOL_TABLE (do not create a symbol table). Use the /SYMBOL_TABLE qualifier when you want the linker to create a symbol table object module file (default file type STB) that contains symbol definitions for all global symbols in the image being linked. The symbol table file can be subsequently specified in LINK commands to provide the symbol definitions to other images.

If you specify /DEBUG, the linker creates a separate symbol table file and it includes within the image the global symbol definitions that are used by the debugger.

When you specify /SYMBOL_TABLE, you can control the defaults applied to the output file specification. Optionally, you may designate a name for the symbol table file, but you may not use wildcard characters.

/SYSLIB
/NOSYSLIB

The default is /SYSLIB (search the system libraries). Use the /NOSYSLIB qualifier to prevent the linking operation from automatically searching the default system libraries, SYS$LIBRARY:IMAGELIB.OLB and then SYS$LIBRARY:STARLET.OLB, for unresolved references in the input files.

/SYSSHR
/NOSYSSHR

The default is /SYSSHR (search the default system shareable image library). Use the /NOSYSSHR qualifier to prevent the linking operation from automatically searching the default system shareable image library, SYS$LIBRARY:IMAGELIB.OLB, for unresolved references. By default, the linker automatically searches the object module library SYS$LIBRARY:STARLET.OLB and then the SYS$LIBRARY:IMAGELIB.OLB library when it cannot resolve references in the input files.

/SYSTEM[=base-address]
/NOSYSTEM

The default is /NOSYSTEM (do not produce a system image). Use the /SYSTEM qualifier to produce a system image and optionally assign it a base address. You cannot use the /SYSTEM qualifier with either the /SHAREABLE qualifier or the /DEBUG qualifier. A system image cannot be run with the RUN command; it must be bootstrapped or otherwise loaded into memory.

The base address specifies where the image is to be loaded in virtual memory It can be expressed in decimal, hexadecimal, or octal format, using the radix specifiers %D, %X, or %O, respectively. The default base address is %X80000000.

System images are intended for special purposes, such as stand-alone operating system diagnostics. When the linker creates a system image, it orders the program sections in alphanumeric order and ignores all program section attributes.

/TRACEBACK (default)
/NOTRACEBACK

Default is /TRACEBACK (include traceback information in the image file to help the system trace the call stack when an error occurs). Use the /NOTRACEBACK qualifier to prevent the linker from including traceback information.

If you specify /DEBUG, /TRACEBACK is assumed.

/USERLIBRARY[=(table[,...])]
/USERLIBRARY=ALL

You use this qualifier to specify which user-defined default libraries (process, group, system or, by default, all three) the linker searches after it has searched any specified user libraries. (The discussion of the linker in the *VMS Linker Utility Manual* explains user-defined default libraries.) You can specify the following tables for the linker to search:

ALL	By default, the linker searches the process, group, and system logical name tables for user-defined library definitions.
GROUP	The linker searches the group logical name table for user-defined library definitions.
NONE	The linker does not search any logical name table; this specification is equivalent to /NOUSERLIBRARY.
PROCESS	The linker searches the process logical name table for user-defined library definitions.
SYSTEM	The linker searches the system logical name table for user-defined library definitions.

If you do not specify either /NOUSERLIBRARY or /USERLIBRARY=(table), the linker assumes /USERLIBRARY=ALL by default.

The /NOUSERLIBRARY qualifier tells the linker not to search any user-defined default libraries.

EXAMPLES

1 `$ LINK ORION`

The LINK command in this example links the object module in the file ORION.OBJ and creates an executable image named ORION.EXE.

2
```
$ LINK/MAP/FULL DRACO,CYGNUS,LYRA
```

The LINK command in this example links the modules DRACO.OBJ, CYGNUS.OBJ, and LYRA.OBJ and creates an executable image named DRACO.EXE. The /MAP and /FULL qualifiers request a full map of the image, with descriptions of each program section, lists of global symbols by name and by value, and a summary of the image characteristics. The map file is named DRACO.MAP.

3
```
$ LINK [SSTEST]SERVICE/INCLUDE=DRACO, -
$_[]CYGNUS/EXECUTABLE
```

The LINK command in this example links the object module DRACO from the library SERVICE.OLB in the directory SSTEST with the module CYGNUS.OBJ in the current default directory. The executable image is named CYGNUS.EXE. The placement of the /EXECUTABLE qualifier provides the output file name default.

4
```
$ LINK/MAP/CROSS_REFERENCE/EXECUTABLE=DBGWEATH -
$_/DEBUG -
$_WEATHER,MATHLIB/LIBRARY
$ RUN DBGWEATH

            VAX DEBUG V4.4

%DEBUG-I-INITIAL, language is FORTRAN, module set to 'WEATHER'
DBG>
```

The LINK command in this example links the object module WEATHER.OBJ with the debugger. If any unresolved references are encountered, the linker searches the library MATHLIB.OLB before searching the system library. The /CROSS_REFERENCE qualifier requests a cross-reference listing in the map file; the map file is named, by default, WEATHER.MAP. The /EXECUTABLE qualifier requests the linker to name the output file DBGWEATH.EXE. The RUN command executes the image; the message from the debugger indicates that it is ready to accept debug commands.

LOGIN Procedure

Initiates an interactive terminal session.

FORMAT

CTRL/C
CTRL/Y
RET

DESCRIPTION

There is no LOGIN command. You signal your intention to access the system by pressing CTRL/C, CTRL/Y, or RET on a terminal not currently in use. The system prompts for your user name and your password (and your secondary password, if you have one) and then validates them.

Specify the optional qualifiers immediately after you type your user name; then press RETURN to get the password prompts.

The login procedure performs the following functions:

- Validates your right to access the system by checking your user name and passwords against the entries in the system's user authorization file
- Establishes the default characteristics of your terminal session based on your user name entry in the authorization file
- Executes the command procedure file SYS$SYLOGIN.COM if one exists
- Executes either the command procedure file named LOGIN.COM if one exists in your default directory, or the command file defined in the user authorization file, if any

Some systems are set up with a retry facility for users who are accessing the system from remote or dialup locations. With these systems, when you make a mistake typing your user name or password, the system allows you to reenter the information without shutting you off. To reenter your login information, press RETURN. The system displays the user name prompt again. Now retype your user name and press RETURN to send the information to the system. The system displays the password prompt. (There is both a limit to the number of times you can retry to enter your login information and a time limit between tries.)

QUALIFIERS

/CLI=command-language-interpreter

Specifies the name of an alternate command language interpreter (CLI) to override the default CLI listed in the user authorization file. The CLI you specify must be located in SYS$SYSTEM and have the file type EXE.

If you do not specify a command interpreter using the /CLI qualifier and do not have a default CLI listed in the user authorization file, the system supplies a default of /CLI=DCL.

/COMMAND[=file-spec]
/NOCOMMAND

Controls whether to execute your default login command procedure when you log in. Use the /COMMAND qualifier to specify the name of an alternate login command procedure. If you specify a file name without a file type, the default file type COM is used. If you specify /COMMAND and omit the file specification, your default login command procedure is executed. By default, /COMMAND is assumed.

Use the /NOCOMMAND qualifier if you do not want your default login command procedure to be executed.

/DISK=device-name[:]

Specifies the name of a disk device to be associated with the logical device SYS$DISK for the terminal session. This specification overrides the default SYS$DISK device established in the authorization file.

/TABLES=(command-table[,...])

Specifies the name of an alternate CLI table to override the default listed in the user authorization file (UAF). This table name is considered a file specification. The default device and directory is SYS$SHARE and the default file type is EXE.

If a logical name is used, the table name specification must be defined in the system logical name table.

If the /CLI qualifier is set to DCL or MCR, the /TABLES qualifier defaults to the correct value. If the /TABLES qualifier is specified without the /CLI qualifier, the CLI specified in the user's UAF will be used.

The default is /TABLES=DCLTABLES.

EXAMPLES

1

```
CTRL/Y
Username:  SMITHSON
Password:
```

In this example, CTRL/Y accesses the operating system, which immediately prompts for a user name. After validating the user name, the system prompts for the password but does not echo it.

2

```
RET
Username:  HIGGINS/DISK=USER$
Password:
       Welcome to VAX/VMS Version 5.00 on node JUPITER
       Last interactive login on Tuesday, 16-AUG-1988 09:16:47.08
       Last non-interactive login on Monday, 15-AUG-1988 17:32:34.27
$ SHOW DEFAULT
  USER$:[HIGGINS]
```

In this example, the /DISK qualifier requests that the default disk for the terminal session be DISK2. The SHOW DEFAULT command shows that USER$ is the default disk.

3 CTRL/C

```
Username: LIZA/CLI=MCR/COMMAND=ALTLOGIN.COM
Password:
       Welcome to VAX/VMS Version 5.00 on node JUPITER
       Last interactive login on Tuesday, 16-AUG-1988 09:16:47.08
       Last non-interactive login on Monday, 15-AUG-1988 17:32:34.27
>
```

In this example, the /CLI qualifier requests the alternate MCR command interpreter. The /COMMAND qualifier indicates that the login command file ALTLOGIN.COM is to be executed instead of the default login command file.

The right angle bracket indicates that MCR is active and expects an MCR command.

4 RET

```
Username: XENAKIS
Password:
Password:
       Welcome to VAX/VMS Version 5.00 on node JUPITER
       Last interactive login on Tuesday, 16-AUG-1988 09:16:47.08
       Last non-interactive login on Monday, 15-AUG-1988 17:32:34.27
$
```

In this example, the second password prompt indicates that the user has a secondary password, which must be entered to access the system.

5 RET

```
Username: JONES
Password:
User authorization failure
RET
Username: JONES
Password:
        Welcome to VAX/VMS Version 5.00 on node JUPITER
        Last interactive login on Tuesday, 16-AUG-1988 09:16:47.08
        Last non-interactive login on Monday, 15-AUG-1988 17:32:34.27
        1 failure since last successful login.
$
```

This example shows the "User authorization failure" message, which indicates that the password has been entered incorrectly. After successfully logging in, a message is displayed showing the number of login failures since your last successful login. This message is displayed only if one or more login failures have occurred.

6 RET

```
Username: JOYCE
Password:
       Welcome to VAX/VMS Version 5.00 on node JUPITER
       Last interactive login on Tuesday, 16-AUG-1988 09:16:47.08
       Last non-interactive login on Monday, 15-AUG-1988 17:32:34.27
       WARNING - Primary password has expired; update immediately.
$
```

This example shows the WARNING message, which indicates that your primary password has expired. You must use the SET PASSWORD command to change your password before logging out, or you will be unable to log in.

For more information on changing your password, see the description of the SET PASSWORD command in this manual.

LOGOUT

Terminates an interactive terminal session.

FORMAT LOGOUT

DESCRIPTION

You must use the LOGOUT command to end a terminal session. Under most circumstances, if you turn the power off at your terminal or hang up your telephone connection without using the LOGOUT command, you remain logged in.

When you use the SET HOST command to log in to a remote processor, you generally need to use the LOGOUT command to end the remote session.

QUALIFIERS

/BRIEF

Prints a brief logout message (process name, date, and time) or a full logout message (a brief message plus accounting statistics).

/FULL

Requests the long form of the logout message. When you specify /FULL, the command interpreter displays a summary of accounting information for the terminal session. The default for a batch job is /FULL.

/HANGUP
/NOHANGUP

For dialup terminals, determines whether or not the phone hangs up whenever you log out. By default, the /HANGUP setting of your terminal port determines whether the line is disconnected. Your system manager determines whether you are permitted to use this qualifier.

EXAMPLES

1

```
$ LOGOUT
  HIGGINS    logged out at 15-APR-1988 17:48:56.73
```

In this example, the LOGOUT command uses the default /BRIEF format. No accounting information is displayed.

2

```
$ LOGOUT/FULL
  HIGGINS    logged out at 15-APR-1988 14:23:45.30
Accounting information:
Buffered I/O count:         22     Peak working set size:     90
Direct I/O count:           10     Peak virtual size:         69
Page faults:                68     Mounted volumes:            0
Charged CPU time: 0 00:01:30.50  Elapsed time:      0 04:59:02.63
```

In this example, the LOGOUT command with the /FULL qualifier displays a summary of accounting statistics for the terminal session.

MACRO

Invokes the VAX MACRO assembler to assemble one or more assembly language source files. For a complete functional description of the VAX MACRO assembler directives, see the *VAX MACRO and Instruction Set Reference Manual.*

This description provides a functional overview of the MACRO command, emphasizing DCL syntax and grammar.

See the qualifier descriptions for restrictions.

FORMAT

MACRO *file-spec-list*

PARAMETER

file-spec-list

Requests the assembly of one or more VAX MACRO assembly language source files. The file-spec-list parameter consists of one or more file specifications. For each file specification, the MACRO command supplies a default file type of MAR. You cannot include a wildcard character in a file specification.

File specifications separated by commas cause the MACRO assembler to produce an object file (and, if indicated, a listing file) for each specified file. File specifications separated by plus signs are concatenated into one input file and produce a single object file (and listing file). The MACRO assembler creates output files of one higher version than the highest version existing in the target directory.

QUALIFIERS

The qualifiers to the MACRO command serve as either command (global) qualifiers or file (positional) qualifiers. A *command qualifier* affects all the files specified in the MACRO command. A *file qualifier* affects only the file that it qualifies. All MACRO qualifiers except the /LIBRARY and /UPDATE qualifiers are usable as either command or file qualifiers. The /LIBRARY and /UPDATE qualifiers are file qualifiers only.

/ANALYSIS_DATA[=file-spec]
/NOANALYSIS_DATA (default)

Controls whether the assembler creates an analysis data file for the VAX Source Code Analyzer (SCA), and optionally provides the file specification.

By default, the assembler does not create an analysis data file. If you specify /ANALYSIS_DATA without a file specification, the assembler creates a file with the same file name as the first input file for the MACRO command. The default file type for analysis data files is ANA. When you specify /ANALYSIS_DATA, you can control the defaults applied to the output file specification by the placement of the qualifier in the command.

/CROSS_REFERENCE[=(function[,...])]
/NOCROSS_REFERENCE (default)

Controls whether a cross-reference listing of the locations in the source file where the specified function (or functions) is defined or referenced. If you

specify more than one function, separate each with a comma and enclose the entire list in parentheses.

You can specify the following functions:

ALL	Cross-references directives, macros, operation codes, registers, and symbols
DIRECTIVES	Cross-references directives
MACROS	Cross-references macros
OPCODES	Cross-references operation codes
REGISTERS	Cross-references registers
SYMBOLS	Cross-references symbols

Because the assembler writes the cross-references to the listing file, you must specify the /LIST qualifier with the /CROSS_REFERENCE qualifier. If you specify no functions in the /CROSS_REFERENCE qualifier, the assembler assumes the default value of /CROSS_REFERENCE=(MACROS,SYMBOLS). The /NOCROSS_REFERENCE qualifier excludes the cross-reference listing.

/DEBUG[=option]
/NODEBUG (default)

Includes or excludes local symbols in the symbol table or traceback information in the object module. You can replace /ENABLE and /DISABLE with /DEBUG and /NODEBUG when you use the appropriate DEBUG and TRACEBACK options. /DEBUG or /NODEBUG override debugging characteristics set with the .ENABLE or .DISABLE assembler directives.

You can specify one or more of the following options:

ALL	Includes in the object module all local symbols in the symbol table, and provides all traceback information for the debugger. This qualifier is equivalent to /ENABLE=(DEBUG,TRACEBACK).
NONE	Makes local symbols and traceback information in the object module unavailable to the debugger. This qualifier is equivalent to /DISABLE=(DEBUG,TRACEBACK).
SYMBOLS	Makes all local symbols in the object module available to the debugger. Makes traceback information unavailable to the debugger. This qualifier is equivalent to /ENABLE=DEBUG and /DISABLE=TRACEBACK together.
TRACEBACK	Makes traceback information in the object module available to the debugger and local symbols unavailable to the debugger. This qualifier is equivalent to /ENABLE=TRACEBACK and /DISABLE=DEBUG together.

If you specify no options to the /DEBUG qualifier, it assumes the default value of /DEBUG=ALL.

/DIAGNOSTICS[=file-spec]
/NODIAGNOSTICS (default)

Creates a file containing assembler messages and diagnostic information. If you omit the file specification, the default file name is the same as the source program; the default file type is DIA.

No wildcard characters are allowed in the file specification.

The diagnostics file is reserved for use with Digital layered products, such as, but not limited to, the VAX Language-Sensitive Editor (LSE).

/DISABLE=(function[,...])
/NODISABLE

Provides initial settings for the functions disabled by the .DISABLE assembler directive. You can specify one or more of the following functions:

ABSOLUTE	Assembles relative addresses as absolute addresses
DEBUG	Includes local symbol table information in the object file for use with the debugger
TRUNCATION	Truncates floating-point numbers (if truncation is disabled, numbers are rounded)
GLOBAL	Assumes undefined symbols to be external symbols
SUPPRESSION	Suppresses listing of unreferenced symbols in the symbol table
TRACEBACK	Provides traceback information to the debugger

If you specify more than one function, separate each with a comma and enclose the list with parentheses. If you specify no functions in the /DISABLE qualifier, it assumes the default value of /DISABLE=(ABSOLUTE,DEBUG,TRUNCATION). The /NODISABLE qualifier has the same effect as not specifying the /DISABLE qualifier, or negates the effects of any /DISABLE qualifiers specified earlier on the command line.

/ENABLE=(function[,...])
/NOENABLE

Provides initial settings for the functions controlled by the .ENABLE assembler directive.

The /NOENABLE qualifier has the same effect as not specifying the /ENABLE qualifier, or negates the effects of any /ENABLE qualifiers specified earlier on the command line. You can specify one or more of the functions as listed in the description of the /DISABLE qualifier, separating each with a comma and enclosing the list in parentheses. If you specify no functions in the /DISABLE qualifier, it assumes the default value of /ENABLE=(GLOBAL,TRACEBACK,SUPPRESSION).

/LIBRARY
/NOLIBRARY

Positional qualifier. The /LIBRARY qualifier cannot be used with the /UPDATE qualifier.

The associated input file to the /LIBRARY qualifier must be a macro library. The default file type is MLB. The /NOLIBRARY qualifier has the same effect as not specifying the /LIBRARY qualifier, or negates the effects of any /LIBRARY qualifiers specified earlier on the command line.

The assembler can search up to 16 libraries, one of which is always STARLET.MLB. This number applies to a particular assembly, not necessarily to a particular MACRO command. If you enter the MACRO command so that more than one source file is assembled, but the source files are assembled *separately*, you can specify up to 16 macro libraries for each separate assembly. More than one macro library in an assembly causes the libraries to be searched in reverse order of their specification.

A macro call in a source program causes the assembler to begin the following sequence of searches:

1. An initial search of the libraries specified with the .LIBRARY directive. The assembler searches these libraries in the reverse order of that in which they were declared.
2. If the macro definition is not found in any of the libraries specified with the .LIBRARY directive, a search of the libraries specified in the DCL MACRO command line (in the reverse order in which they were specified).
3. If the macro definition is not found in any of the libraries specified in the command line, a search of STARLET.MLB.

/LIST[=file-spec]
/NOLIST

Creates or omits an output listing, and optionally provides an output file specification for it. The default file type for the listing file is LIS. No wildcard characters are allowed in the file specification.

An interactive MACRO command does not produce a listing file by default. /NOLIST, present either explicitly or by default, causes errors to be reported on the current output device.

/LIST is the default for a MACRO command in a batch job. /LIST allows you to control the defaults applied to the output file specification by the placement of the qualifier in the command. See Section 1.3 of the *VMS DCL Concepts Manual* for more information on entering output file qualifiers.

/OBJECT[=file-spec]
/NOOBJECT

Creates or omits an object module. It also defines the file specification. By default, the assembler creates an object module with the same file name as the first input file. The default file type for object files is .OBJ. No wildcard characters are allowed in the file specification.

/OBJECT controls the defaults applied to the output file specification by the placement of the qualifier in the command. See Section 1.3 of the *VMS DCL Concepts Manual* for more information on entering output file qualifiers.

/SHOW[=(function[,...])]
/NOSHOW[=(function[,...])]

Provides initial settings for the functions controlled by the assembler directives .SHOW and .NOSHOW.

You can specify one or more of the following functions:

CONDITIONALS	Lists unsatisfied conditional code associated with .IF and .ENDC MACRO directives
CALLS	Lists macro calls and repeat range expansions
DEFINITIONS	Lists macro definitions
EXPANSIONS	Lists macro expansions
BINARY	Lists binary code generated by the expansion of macro calls

If you specify more than one function, separate each with a comma and enclose the list in parentheses. If you specify no functions in the /SHOW qualifier, it increments the listing level count; the /NOSHOW qualifier decrements the count in similar circumstances. Because these qualifiers contribute to the listing file, you must also specify the /LIST qualifier when you use them. If you do not specify the /SHOW qualifier, the MACRO command assumes a default of /SHOW=(CONDITIONALS,CALLS,DEFINITIONS). Separate multiple functions by commas and enclose the list in parentheses.

/UPDATE[=(update-file-spec[,...])]
/NOUPDATE

Positional qualifier. The /UPDATE qualifier cannot be used with the /LIBRARY qualifier.

Updates the input file it qualifies using the SLP batch editor and the specified update file or files. By default, the assembler assumes that the update file has the same file name as the input source file and a file type of UPD. You cannot include a wildcard character in the file specifications. If it cannot find a specified update file, the assembler prints an informational message and continues the assembly.

If you specify more than one update file specification, separating each with a comma and enclosing the list in parentheses, the assembler merges the contents into a single list of updates before applying the updates to the source file.

The /NOUPDATE qualifier has the same effect as not specifying the /UPDATE qualifier, or negates any /UPDATE qualifiers specified earlier on the command line. The input source file and update files are not changed by the update operation. The effects of the update appear in the compiled output. If you specify the /LIST qualifier with the /UPDATE qualifier, the assembler writes an audit trail of the changes to the listing file.

EXAMPLES

1
```
$ MACRO/LIST  CYGNUS, LYRA/OBJECT=LYRAN + MYLIB/LIBRARY
```

In this example, the MACRO command requests two separate assemblies. Using .MAR as the default, MACRO assembles CYGNUS.MAR to produce CYGNUS.LIS and CYGNUS.OBJ. Then it assembles LYRA.MAR and creates a listing file named LYRA.LIS and an object module named LYRAN.OBJ. The default output file type for a listing is .LIS.

The command requests the search of the MYLIB library file in the current directory for macro definitions.

2 `$ MACRO  ORION`

MACRO assembles the file ORION.MAR and creates an object file named ORION.OBJ. Executing the command in a batch job causes MACRO to create a listing file named ORION.LIS.

3
```
$ MACRO  ALPHA/LIST+MYLIB/LIBRARY-
$_ + [TEST]OLDLIB/LIBRARY + []BETA
$ PRINT ALPHA
```

MACRO concatenates the files ALPHA.MAR and BETA.MAR to produce an object file named ALPHA.OBJ and a listing file named ALPHA.LIS. The command line requests the search of libraries MYLIB.MLB (in the current default directory) and OLDLIB.MLB (in the directory [TEST]) for macro definitions. When macro calls are found in BETA.MAR, MACRO searches the libraries OLDLIB, MYLIB, and the system library STARLET.MLB, in that order, for the definitions.

The PRINT command prints the listing file ALPHA.LIS.

4 `$ MACRO  DELTA+TESTLIB/LIBRARY, ALPHA+MYLIB/LIBRARY`

MACRO requests two separate assemblies. MACRO searches TESTLIB.MLB and the system library STARLET.MLB for macro definitions when macro calls are found in DELTA.MAR, and searches MYLIB.MLB and the system library STARLET.MLB for macro definitions when macro calls are found in ALPHA.MAR.

MAIL

Invokes the Personal Mail Utility (MAIL), which is used to send messages to other users of the system. For a complete description of the Personal Mail Utility, including information about the MAIL command and its qualifiers, see the *VMS Mail Utility Manual*.

FORMAT

MAIL *[file-spec] [recipient-name]*

MERGE

Invokes the Sort/Merge Utility to combine two through ten similarly sorted input files and create a single output file. Note that input files to be merged must be in sorted order. For a complete description of the Sort/Merge Utility, including more information about the MERGE command and its qualifiers, see the *VMS Sort/Merge Utility Manual*.

FORMAT

MERGE *input-file-spec1,input-file-spec2[,...] output-file-spec*

MESSAGE

Invokes the Message Utility (MESSAGE) to compile one or more files of message definitions. For a complete description of the Message Utility, including more information about the MESSAGE command and its qualifiers, see the *VMS Message Utility Manual*.

FORMAT MESSAGE *file-spec[,...]*

MONITOR

Invokes the Monitor Utility (MONITOR) to monitor classes of systemwide performance data at a specified interval. For a complete description of the Monitor Utility, including information about the MONITOR command, refer to the *VMS Monitor Utility Manual*.

FORMAT

MONITOR *[class-name[,...]]*

MOUNT

Invokes the Mount Utility (MOUNT) to make a disk or magnetic tape volume available for processing. For a complete description of the Mount Utility, including information about the MOUNT command and its qualifiers, see the *VMS Mount Utility Manual*.

FORMAT

MOUNT *device-name[:][,...] [volume-label[,...]] [logical-name[:]]*

NCS

Invokes the VMS National Character Set Utility to provide a convenient method of implementing alternative (non-ASCII) string collating sequences, typically using subsets of the Multinational Character Set. NCS also facilitates the implementation of string conversion functions.

For a complete description of the NCS utility, including more information about the NCS command, see the *VMS National Character Set Utility Manual*.

FORMAT

NCS *[file-spec,...]*

ON

Performs a specified action when a command or program executed within a command procedure encounters an error condition or is interrupted by CTRL/Y. The specified actions are performed only if the command interpreter is enabled for error checking or CTRL/Y interrupts (the default conditions). Use the ON command only in a command procedure.

FORMAT

ON *condition THEN [$] command*

PARAMETERS

condition

Either the severity level of an error or a CTRL/Y interrupt. Specify one of the following keywords, which may be abbreviated to one or more characters:

WARNING	Return status of warning occurs ($SEVERITY equals 0)
ERROR	Return status of error occurs ($SEVERITY equals 2)
SEVERE_ERROR	Return status of error occurs ($SEVERITY equals 4)
CONTROL_Y	CTRL/Y character occurs on SYS$INPUT

The default error condition is ON ERROR THEN EXIT.

To specify a CRTL/Y interrupt, use the following keyword:

CONTROL/Y

command

The DCL command line to be executed. You can optionally precede the command line with a dollar sign ($).

If you specified an error condition as the condition parameter, the action is taken when errors equal to or greater than the specified level of error occur.

DESCRIPTION

During the execution of a command procedure, the command interpreter checks the condition code returned from each command or program that executes. With the ON command, you can establish a course of action for the command interpreter to take based on the result of the check.

The system places condition codes in the global symbol $STATUS. The severity of the condition code is represented in the first three low-order bits of $STATUS. This severity level is also represented by the global symbol $SEVERITY. See the description of the EXIT command for information on severity level values.

If an ON command action specifies the severity level of an error, the command interpreter executes the ON command action for errors at the specified severity level or greater. For example, the following command causes a procedure to exit on warnings, errors, or severe errors:

```
$ ON WARNING THEN EXIT
```

The default action is as follows:

```
$ ON ERROR THEN EXIT
```

That is, the command interpreter continues when a warning occurs, and executes an EXIT command when an error or severe error occurs. An ON command action that specifies a severity level is executed only once; after the ON command action is taken, the default ON action is reset. There is an exception to the default ON action. If you use the GOTO command and specify a label that does not exist in the current command procedure, the procedure issues a warning message and exits.

The action specified by an ON command applies only within the command procedure in which the command is executed. Therefore, if you execute an ON command in a procedure that calls another procedure, the ON command action does not apply to the nested procedure. An ON command executed at any command procedure level does not affect the error condition handling of procedures at any other level.

To disable error checking with the ON command, use the SET NOON command. You can enable error checking with the SET ON command, or by entering another ON command.

The ON command also provides a way to define an action routine for a CTRL/Y interrupt that occurs during execution of a command procedure. The default CTRL/Y action is to prompt for command input at the CTRL/Y command level. The CTRL/Y command level is a special command level where you can enter DCL commands. If you enter a command that is executed within the command interpreter, you can resume execution of the procedure with the CONTINUE command. (See Table 1–1 in the *VMS DCL Concepts Manual* for a list of commands that are executed within the command interpreter.)

If you enter any other DCL command, the command interpreter returns to command level 0 and executes the image invoked by the command. If you interrupt the command procedure while it is executing an image that contains an exit handler, the exit handler is allowed to execute before the new command (image) is executed. (However, if you enter the STOP command after you interrupt a command procedure with CTRL/Y, exit handlers declared by the interrupted image are not executed.)

You can use the ON command to change the default action for a CTRL/Y interrupt. If you change the default CTRL/Y action, the execution of a CTRL/Y does not automatically reset the default CTRL/Y action. A CTRL/Y action remains in effect until one of the following occurs:

- The procedure terminates (as a result of an EXIT or STOP command, or as a result of a default error condition handling action)
- Another ON CONTROL_Y command is executed
- The procedure executes the SET NOCONTROL=Y command

A CTRL/Y action can be specified for each active command level; the CTRL/Y action specified for the currently executing command level overrides actions specified for previous levels.

Note: The ON CONTROL_Y and SET NOCONTROL=Y commands are intended for special applications. It is not recommended, in general, that you disable CTRL/Y interrupts. For example, if a procedure that

disables CTRL/Y interrupts begins to loop uncontrollably, you cannot gain control to stop the procedure from your terminal.

EXAMPLES

1
```
$ ON SEVERE_ERROR THEN CONTINUE
```

A command procedure that contains this statement continues to execute normally when a warning or error occurs during execution. When a severe error occurs, the ON statement signals the procedure to execute the next statement anyway. Once the statement has been executed as a result of the severe error condition, the default action (ON ERROR THEN EXIT) is reinstated.

2
```
$ ON ERROR THEN GOTO BYPASS $ RUN A $ RUN B
   .
   .
   .
$ EXIT
$ BYPASS:
$      RUN C
```

If either program A or program B returns a status code with a severity level of error or severe error, control is transferred to the statement labeled BYPASS and program C is run.

3
```
$ ON WARNING THEN EXIT
   .
   .
   .
$ SET NOON
$ RUN [SSTEST]LIBRA
$ SET ON
   .
   .
   .
```

The ON command requests that the procedure exit when any warning, error, or severe error occurs. Later, the SET NOON command disables error checking before executing the RUN command. Regardless of any status code returned by the program LIBRA.EXE, the procedure continues. The next command, SET ON, reenables error checking and reestablishes the most recent ON condition.

4
```
$ ON CONTROL_Y THEN GOTO CTRL_EXIT
   .
   .
   .
$ CTRL_EXIT:
$ CLOSE INFILE
$ CLOSE OUTFILE
$ EXIT
```

The ON command specifies action to be taken when CTRL/Y is pressed during the execution of this procedure: the GOTO command transfers control to the line labeled CTRL_EXIT. At CTRL_EXIT, the procedure performs clean-up operations (in this example, closing files and exiting).

OPEN

Opens a file for reading and/or writing, assigns a logical name to a file, and places the name in the process logical name table.

See the qualifier descriptions for restrictions.

FORMAT

OPEN *logical-name[:] file-spec*

PARAMETERS

logical-name[:]

Specifies the logical name and a character string to be assigned to the file.

file-spec

Specifies the name of the file or device being opened for input or output. The file type defaults to DAT. Wildcard characters are not allowed.

To create a new, sequential file, specify the /WRITE qualifier. See the description of the /WRITE qualifier for more information.

DESCRIPTION

A file can be opened for either reading or writing, or for both reading and writing. After the file is opened, it is available for input or output at the command level with the READ and WRITE commands.

The OPEN command opens files as process-permanent. Therefore, these files remain open until you close them with the CLOSE command, or until you log out. If a command procedure that opens a file terminates without closing an open file, the file remains open; the command interpreter does not automatically close it. The OPEN command uses VAX-11 RMS to open files, and is subject to RMS restrictions on using process permanent files. The OPEN command opens sequential, relative, or indexed sequential files.

The logical devices SYS$INPUT, SYS$OUTPUT, SYS$COMMAND, and SYS$ERROR do not have to be explicitly opened before they can be read or written at the command level. All other files must be explicitly opened.

Do not use the same logical name when you open different files. If you specify a logical name with the OPEN command and the logical name is currently assigned to another file, no warning message is issued. However, the file is not opened, and the next READ request will reference the file to which the logical name was originally assigned.

You can enter more than one OPEN command for the same file and assign it different logical names if you use the /SHARE qualifier the first time the file is opened. Also, if you open the file using the /SHARE=READ or /SHARE=WRITE qualifier, other users may access the file with the TYPE or SEARCH command.

QUALIFIERS

/APPEND

Opens an existing file for writing and positions the record pointer at the end-of-file. New records are added to the end of the file.

Use the /APPEND qualifier only to add records to an existing file. The /APPEND and the /WRITE qualifiers are mutually exclusive.

/ERROR=label

Transfers control to the location specified by the label keyword (in a command procedure) if the OPEN operation results in an error. The error routine specified for this qualifier overrides any ON condition action specified. If /ERROR is not specified, the current ON condition action is taken.

If an error occurs and the target label is successfully given control, the global symbol $STATUS retains the code for the error that caused the error path to be taken.

/READ (default)

Opens the file for reading. If you specify the /READ qualifier without the /WRITE qualifier, you must specify an existing file.

/SHARE[=option]

Opens the specified file as a shareable file to allow other users read or write access. If you specify /SHARE=READ, users are allowed read access to the file. If you specify /SHARE=WRITE or omit the option, users are allowed read and write access to the specified file.

/WRITE

Opens the file for writing. The following restrictions apply to the /WRITE qualifier:

- Use the /WRITE qualifier to open and create a new, sequential file. If the file specification on an OPEN/WRITE command does not include a file version number, and if a file with the specified file name and file type already exists, a new file with a version number one greater than the existing file is created.
- Use the /READ qualifier with the /WRITE qualifier to open an existing file. When the file is opened, the pointer is positioned to the beginning of the file. (This differs from OPEN/APPEND, which positions the pointer at the end of the file.) You cannot use OPEN/READ/WRITE to create a new file.
- The /WRITE and the /APPEND qualifiers are mutually exclusive.

EXAMPLES

1

```
$ OPEN INPUT_FILE AVERAGE.DAT
$ READ_LOOP:
$ READ/END_OF_FILE=ENDIT  INPUT_FILE  NUM
   .
   .
   .
$ GOTO READ_LOOP
$ ENDIT:
$ CLOSE INPUT_FILE
```

The OPEN command opens the file named AVERAGE.DAT as an input file and assigns it the logical name INPUT_FILE. The file is opened with read access because the /READ qualifier is present by default. The READ command reads a record from the logical file INPUT_FILE into the symbol named NUM. The procedure executes the lines between the labels READ_LOOP and ENDIT until the end of the file is reached. At the end of the file, the CLOSE command closes the file.

2

```
$ OPEN/WRITE/ERROR=OPEN_ERROR  OUTPUT_FILE  TEMP.OUT
$ COUNT = 0
$ WRITE_LOOP:
$ COUNT = COUNT +1
$ IF COUNT .EQ. 11 THEN GOTO ENDIT
$ WRITE OUTPUT_FILE "Count is ''COUNT'."
   .
   .
   .
$ GOTO WRITE_LOOP
$ ENDIT:
$ CLOSE OUTPUT_FILE
$ EXIT
$
$ OPEN_ERROR:
$ WRITE SYS$OUTPUT "Cannot open file TEMP.OUT"
$ EXIT
```

The OPEN command with the /WRITE qualifier creates the file TEMP.OUT and assigns it the logical name OUTPUT_FILE. TEMP.OUT is a sequential file.

The /ERROR qualifier specifies that if any error occurs while opening the file, the command interpreter should transfer control to the line at the label OPEN_ERROR. The command procedure writes records to the file TEMP.OUT until the symbol COUNT equals 11.

3

```
$ OPEN/READ INPUT_FILE TRNTO::DBAO:[COST]INVENTORY.DAT
$ READ_LOOP:
$ READ/END_OF_FILE=ENDIT  INPUT_FILE  NUM
$ FIRST_CHAR = F$EXTRACT(0,1,NUM)
$ WRITE SYS$OUTPUT FIRST_CHAR
$ GOTO READ_LOOP
$ ENDIT:
$ CLOSE INPUT_FILE
```

This command procedure opens the file INVENTORY.DAT located at remote node TRNTO as an input file, and assigns it the logical name INPUT_FILE. The READ command reads a record from the logical file INPUT_FILE into the symbol named NUM. The next two commands extract the first character from the record and write the character to the SYS$OUTPUT device. These two steps occur for all records in the file until the procedure reaches the end-of-file. At this point, the CLOSE command closes the file and deassigns the logical name INPUT_FILE.

PASSWORD

When submiting a batch job through a card reader, provides the password associated with the user name that is specified with the JOB card. Although the PASSWORD card is required, the password on the card is optional if the account has a null password.

The PASSWORD command is valid only in a batch job submitted through a card reader and requires that a dollar sign precede the PASSWORD command on the card.

FORMAT

$ PASSWORD *[password]*

PARAMETER

password

Specifies the password associated with the user name specified with the JOB command. *Password* can be 1 to 31 characters long.

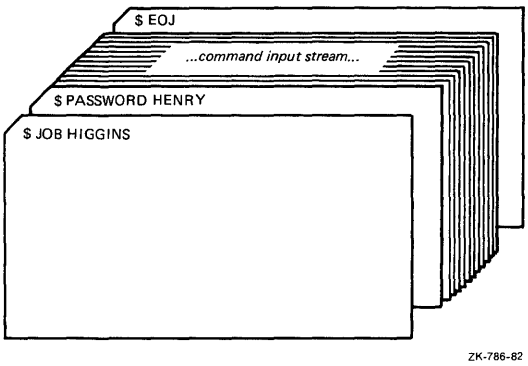
If you are submitting the job from an account with a null password, omit the password specifier on the PASSWORD card.

DESCRIPTION

The PASSWORD command is used in conjunction with the JOB command. The JOB card identifies the user submitting the batch job through a card reader and is followed by a PASSWORD card giving the password. The password is checked by the system to verify that it matches the password associated with the user name on the JOB card. If the passwords do not match, the job is rejected.

Note that you might want to suppress printing when you originally keypunch the PASSWORD card to prohibit other users from seeing the password when the PASSWORD card is in use.

EXAMPLE



The JOB and PASSWORD commands precede a batch job submitted from the card reader. An EOJ command marks the end of the job.

PATCH

Invokes the Image File Patch Utility (PATCH) to patch an executable image, shareable image, or device driver image. For a complete functional description of the Patch Utility, including more information about the PATCH command and its qualifiers, see the *VMS Patch Utility Manual*.

FORMAT

PATCH *file-spec*

PHONE

Invokes the Phone Utility that allows you to communicate with other users on your system or any other VMS system connected to your system by DECnet–VAX. For a complete description of the Phone Utility, including information on the PHONE command and its qualifiers, see the *VMS Phone Utility Manual*.

FORMAT

PHONE *[phone-command]*

PRINT

Queues one or more files for printing, either to the default system printer queue or to a specified queue.

Requires the /REMOTE qualifier if you include a node name in your file specification. Requires OPER privilege, EXECUTE (E) access to the queue, or WRITE (W) access to the queue.

FORMAT

PRINT *file-spec[,...]*

PARAMETER

file-spec[,...]

Specifies one or more files to be printed. Either commas or plus signs can be used to separate file specifications. Wildcard characters are allowed in the directory specification, file name, file type, or version number fields.

The PRINT command concatenates the files into a single print job and by default, gives the job the name of the first file specified. If you do not specify a file type for the first input file, the PRINT command uses the default file type LIS.

Node names are allowed only when the /REMOTE qualifier is used. The file must not reside on an allocated device.

DESCRIPTION

The PRINT command places the specified files in a printer or terminal queue for printing. All files queued by a single PRINT command are considered one job. The system assigns a unique job identification number—the "job entry number"—to each job in the queue and displays this entry number when the PRINT command completes execution. (Completion of a PRINT command means that the job has been queued to the appropriate printer queue. It does not imply that the actual printing process is finished.)

Once a print job has been queued, the version of the file submitted is printed, even if a newer version of the file is created before the print job runs.

Printer queues are identified by name. If you do not specify a queue name with the /QUEUE qualifier, the system queues the job to SYS$PRINT. The PRINT command, by default, displays the name of the queue on which it entered the job.

QUALIFIERS

/AFTER=time
/NOAFTER

Holds the job until the specified time. The time can be specified as an absolute time or a combination of absolute and delta times. See Section 1.4 of the *VMS DCL Concepts Manual* for complete information on specifying time values. If the specified time has passed, the job is queued for printing immediately.

/BACKUP
/NOBACKUP

Modifies the time value specified with the /BEFORE or /SINCE qualifier. /BACKUP selects files according to the dates of their most recent backups. This qualifier is incompatible with the other qualifiers that also allow you to select files according to time attributes: /CREATED, /EXPIRED, and /MODIFIED. If you specify none of these four time qualifiers, the default is /CREATED.

/BEFORE[=time]
/NOBEFORE

Selects only those files dated prior to the specified time. You can specify time as an absolute time, as a combination of absolute and delta times, or as one of the following keywords: TODAY (default), TOMORROW, or YESTERDAY. Specify one of the following qualifiers with /BEFORE to indicate the time attribute to be used as the basis for selection: /BACKUP, /CREATED (default), /EXPIRED, or /MODIFIED.

See Section 1.4 of the *VMS DCL Concepts Manual* for complete information on specifying time values.

/BURST[=keyword]
/NOBURST

Positional qualifier.

Controls whether a burst page is printed preceding a file. (A burst page is a flag page printed over the perforation between pages for easy identification of individual files.) If the /BURST qualifier is specified between the PRINT command and the file specifications, it can take either of two keywords:

ALL	Prints a burst page before each file in the job
ONE	Prints a burst page before the first file in the job

If you want the /BURST qualifier to apply to individual files in a multifile job, place the qualifier directly after each file that you want to have a burst page. The default is /NOBURST.

Use the /[NO]BURST qualifier to override the installation-defined defaults that have been set for the printer queue you are using; these defaults are established with the INITIALIZE/QUEUE command.

When you specify /BURST, you need not specify /FLAG; a flag page automatically follows a burst page.

/BY_OWNER[=uic]
/NOBY_OWNER

Selects only those files whose owner user identification code (UIC) matches the specified owner UIC. The default UIC is that of the current process.

Specify the UIC using standard UIC format as described in Section 8.1 of the *VMS DCL Concepts Manual.*

/CHARACTERISTICS=(characteristic[,...])

Specifies one or more characteristics desired for printing the files. If you specify only one characteristic, you can omit the parentheses. Characteristics can refer to such things as color of ink. Codes for characteristics can be either names or values from 0 to 127 and are installation-defined.

Use the SHOW QUEUE/CHARACTERISTICS command to see which characteristics have been defined for your system (defined with the DEFINE /CHARACTERISTIC command). Use the SHOW QUEUE/FULL command to see which characteristics are available on a particular queue.

A print job can execute on a printer queue only if each characteristic specified with the PRINT command is also specified for that particular printer queue. If you specify a characteristic that has not been specified for that particular printer queue, the job remains pending. (In order for your job to print, the system manager should stop the queue, physically change the characteristics of the printer, and restart the queue, specifying the new values listed in the /CHARACTERISTICS qualifier.)

Specification of a characteristic for a printer queue does not prevent jobs that do not specify that characteristic from being executed.

/CONFIRM
/NOCONFIRM (default)

Controls whether a request is issued before each print operation to confirm that the operation should be performed on that file. The following responses are valid:

YES	NO	QUIT
TRUE	FALSE	CTRL/Z
1	0	ALL
	RET	

You can use any combination of uppercase and lowercase letters for word responses. Word responses can be abbreviated to one or more letters (for example, T, TR, or TRU for TRUE), but these abbreviations must be unique. Affirmative answers are YES, TRUE, and 1. Negative answers are NO, FALSE, 0, and <RET>. QUIT or CTRL/Z indicates that you want to stop processing the command at that point. When you respond with ALL, the command continues to process, but no further prompts are given. If you type a response other than one of those in the list, DCL issues an error message and redisplays the prompt.

/COPIES=n

Positional qualifier.

Specifies the number of copies to print. The value of n can be from 1 to 255 and defaults to 1. If you place the /COPIES qualifier after the PRINT command name, each file in the parameter list is printed the specified number of times. If you specify /COPIES following a file specification, only that file is printed the specified number of times.

/CREATED (default)
/NOCREATED

Modifies the time value specified with the /BEFORE or /SINCE qualifier. /CREATED selects files based on their dates of creation. This qualifier is incompatible with the other qualifiers that also allow you to select files according to time attributes: /BACKUP, /EXPIRED, and /MODIFIED. If you specify none of these four time qualifiers, the default is /CREATED.

/DELETE
/NODELETE (default)
Positional qualifier.

Controls whether files are deleted after printing. If you place the /DELETE qualifier after the PRINT command name, all specified files are deleted. If you specify /DELETE after a file specification, only that file is deleted after it is printed.

The protection applied to the file must allow delete access to the current UIC.

/DEVICE=queue-name[:]
Places the print job in the specified queue (rather than the default queue SYS$PRINT). This qualifier is synonymous with /QUEUE, except that the /DEVICE qualifier is reserved for special use by DIGITAL. Its usage, therefore, is not recommended.

/EXCLUDE=(file-spec[,...])
/NOEXCLUDE
Excludes the specified files from the PRINT operation. You can include a directory but not a device in the file specification. Wildcard characters are allowed in the file specification. However, you cannot use relative version numbers to exclude a specific version. If you provide only one file specification, you can omit the parentheses.

/EXPIRED
/NOEXPIRED
Modifies the time value specified with the /BEFORE or /SINCE qualifier. /EXPIRED selects files according to their expiration dates. (The expiration date is set with the SET FILE/EXPIRATION_DATE command.) The /EXPIRED qualifier is incompatible with the other qualifiers that also allow you to select files according to time attributes: /BACKUP, /CREATED, and /MODIFIED. If you specify none of these four time qualifiers, the default is /CREATED.

/FEED (default)
/NOFEED
Positional qualifier.

Automatically inserts form feeds when pages are within 4 lines of the end of the page (line 62 on 66-line forms). You can reset the number of lines per form with the /FORM qualifier. You can suppress this automatic form feed (without affecting any of the other carriage control functions that are in place) by using the /NOFEED qualifier. The /[NO]FEED qualifier does not affect user-formatted files and can be used to override the installation-defined defaults that have been set for the printer queue you are using .

/FLAG[=keyword]
/NOFLAG
Positional qualifier.

Controls whether a flag page is printed preceding a file. The flag page contains the name of the user submitting the job, the job entry number, and other information about the file being printed. If the /FLAG qualifier is positioned between the PRINT command and the file specifications, it can take either of two keywords:

ALL	Prints a flag page before each file in the job
ONE	Prints a flag page before the first file in the job

If you want the /FLAG qualifier to apply to individual files in a multifile job, place the qualifier directly after each file that you want to have a flag page.

Use the /[NO]FLAG qualifier to override the installation-defined defaults that have been set for the printer queue you are using.

/FORM=type

Specifies the name or number of the form for the print queue (defined with the DEFINE/FORM command). The default is /FORM=0.

Specify the forms type using a numeric value or alphanumeric name. Form types can refer to the print image width and length or the type of paper. Codes for form types are installation-defined. You can use the SHOW QUEUE/FORM command to display the form types available for your system. Use the SHOW QUEUE/FULL command to find out which form is available for a particular queue.

If you specify a form with a stock type different from the stock type of the form on the queue, your job remains pending until the stock type on the queue is set equal to the stock type of the job.

/HEADER
/NOHEADER (default)

File-spec qualifier.

Controls whether a heading line is printed at the top of each page.

/HOLD
/NOHOLD (default)

Controls whether a job is available for printing immediately. The /HOLD qualifier holds the job until released by a SET QUEUE/ENTRY/RELEASE command.

/IDENTIFY (default)
/NOIDENTIFY

Displays the queue name and job number of the job when it is queued.

/JOB_COUNT=n

Prints the job n times. The value of n can be from 1 through 255 and defaults to 1.

/LOWERCASE
/NOLOWERCASE (default)

Indicates whether the job must be printed on a printer that supports both lowercase and uppercase letters.

/MODIFIED
/NOMODIFIED

Modifies the time value specified with the /BEFORE or /SINCE qualifier. /MODIFIED selects files according to the dates on which they were last modified. This qualifier is incompatible with the other qualifiers that also allow you to select files according to time attributes: /BACKUP, /CREATED,

and /EXPIRED. If you specify none of these four time modifiers, the default is /CREATED.

/NAME=job-name

Names the job. The name consists of 1 through 39 alphanumeric characters. The default is the name of the first (or only) file in the job. The job name is used in the SHOW QUEUE command display and is printed on the flag page for the job.

/NOTE=string

Specifies a message string of up to 255 characters to appear on the flag page of the job.

/NOTIFY
/NONOTIFY (default)

Controls whether a message is broadcast to your terminal when the job is printed.

/OPERATOR=string

Specifies a message of up to 255 characters to be sent to the operator when the job begins to print.

/PAGES=([lowlim,]uplim)

Positional qualifier.

Specifies the number of pages to print for the specified job. You can use the /PAGES qualifier to print portions of long files. By default, all pages of the file are printed.

The lowlim specifier refers to the first page in the group of pages that you want printed for that file. If you omit the lowlim specifier, the printing starts on the first page of the file.

The uplim specifier refers to the last page of the file that you want printed. If you want to print to the end of the file, but do not know how many pages the file contains, use ("") as the uplim specifier. You can omit the parentheses if you are including only a specific value for the uplim specifier. For example, /PAGES=10 prints the first ten pages of the file; /PAGES=(5,10) prints pages five through 10; /PAGES=(5,("")) starts printing at page 5 in the file and continues until the end of the file is reached.

/PARAMETERS=(parameter[,...])

Specifies from one to eight optional parameters to be passed to the job; each parameter can contain up to 255 characters. The commas delimit individual parameters. Enclose parameters containing any special characters or delimiters with quotation marks ("").

If you specify only one parameter, you can omit the parentheses.

/PASSALL
/NOPASSALL (default)

Positional qualifier.

Specifies whether the symbiont bypasses all formatting and sends the output QIO to the driver with format suppressed. All qualifiers affecting formatting, as well as the /HEADER, /PAGES, and /SETUP qualifiers, are ignored.

If the /PASSALL qualifier is placed between the PRINT command and any file specifications, all files are printed in PASSALL mode. To specify /PASSALL with only some files in the job, place the qualifier after each file that you want printed in PASSALL mode.

/PRIORITY=n

Requires OPER or ALTPRI privilege to raise the priority above the SYSGEN parameter MAXQUEPRI.

Specifies the priority of the print job. The value of n can be from 0 through 255, where 0 is the lowest priority and 255 is the highest. The default value of n is the value of the SYSGEN parameter DEFQUEPRI. No privilege is needed to set the priority lower than the MAXQUEPRI value.

/QUEUE=queue-name[:]

Queues the job to the specified print queue. If this qualifier is omitted, the default is SYS$PRINT. This qualifier is synonymous with /DEVICE.

/REMOTE

Queues the job to SYS$PRINT on the remote node specified in the file specification; the file *must* exist on the remote node. You can only specify the following qualifiers with /REMOTE: /BACKUP, /BEFORE, /BY_OWNER, /CONFIRM, /CREATED, /EXCLUDE, /EXPIRED, /MODIFIED, and /SINCE.

Note that, unlike the printing on the local node, multiple files queued by a single PRINT/REMOTE command are considered separate jobs.

Not all PRINT qualifiers are compatible with /REMOTE. Only the following qualifiers may be specified with /REMOTE: /BACKUP, /BEFORE, /BY_OWNER, /CONFIRM, /CREATED, /EXCLUDE, /EXPIRED, /MODIFIED, and /SINCE.

/RESTART (default)
/NORESTART

Restarts the job after a crash or a STOP/REQUEUE command.

/SETUP=module[,...]

Extracts the specified modules from the device control library (containing escape sequence modules for programmable printers) and copies the modules to the printer before a file is printed. By default, no device control modules are copied.

Note that the module names are not checked for validity until the time that the file is actually printed. Therefore, PRINT/SETUP is susceptible to typing errors and other mistakes. It is recommended only for experimental setups.

For production setups, see DEFINE/FORM/SETUP.

/SINCE[=time]
/NOSINCE

Selects only those files dated after the specified time. You can specify time as an absolute time, a combination of absolute and delta times, or as one of the following keywords: TODAY (default), TOMORROW, or YESTERDAY. Specify one of the following qualifiers with /BEFORE to indicate the time attribute to be used as the basis for selection: /BACKUP, /CREATED (default), /EXPIRED, or /MODIFIED.

See Section 1.4 of the *VMS DCL Concepts Manual* for complete information on specifying time values.

/SPACE
/NOSPACE (default)

Positional qualifier.

Controls whether print job output is double-spaced. The default is single-spaced output.

/TRAILER[=keyword]
/NOTRAILER

Positional qualifier.

Controls whether a trailer page is printed at the end of a file. The trailer page displays the job entry number as well as information about the user submitting the job and the files being printed. If the /TRAILER qualifier is positioned between the PRINT command and the file specifications, it can take either of two keywords:

ALL	Prints a trailer page after each file in the job
ONE	Prints a trailer page after the last file in the job

If you want the /TRAILER qualifier to apply to individual files in a multifile job, place the qualifier directly after each file that you want to have a trailer page.

Use the /[NO]TRAILER qualifier to override the installation-defined defaults that have been set for the printer queue you are using.

/USER=username

Requires the CMKRNL privilege and R (READ) access to the system authorization file.

Allows you to print a job on behalf of another user. The print job runs exactly as if that user had submitted it. The print job runs under that user's user name and UIC and accounting information is logged to that user's account. By default, the user identification comes from the requesting process. The user name keyword can be any user name that is validated on your system.

EXAMPLES

1
```
$ PRINT/QUEUE=LPB0/COPIES=10/AFTER=20   RESUME
  Job RESUME (queue LPB0 entry 239) holding until 31-DEC-88 20:00
```

The PRINT command in this example queues 10 copies of the file RESUME.LIS to printer LPB0, but requests that the copies not be printed until after 8:00 P.M.

2
```
$ PRINT  ALPHA.TXT + BETA/FLAG + GAMMA/FLAG + *.LIS/FLAG
  Job ALPHA (queue SYS$PRINT, entry 237) pending
```

The PRINT command in this example submits the files ALPHA.TXT, BETA.TXT, GAMMA.TXT, and the highest versions of all files with the file type LIS as a single print job. Flag pages separate the individual files. Notice that the file type for BETA and GAMMA is TXT, the file type of the first file in the list.

3
```
$ PRINT/LOWERCASE   ALPHA.TXT/COPIES=2, -
_$BETA.DOC/COPIES=3
  Job ALPHA (queue SYS$PRINT, entry 240) pending
```

The print job queued by the PRINT command in this example consists of two copies of ALPHA.TXT followed by three copies of BETA.DOC. This job must be printed on a printer that can print lowercase letters. If no such printer is available, the job waits in the queue.

4
```
$ PRINT/JOB_COUNT=3   ALPHA.TXT,BETA/NOIDENTIFY
```

The PRINT command in this example concatenates the files ALPHA.TXT and BETA.TXT into a single print job and prints three copies of the job. The /NOIDENTIFY qualifier requests that the job entry number and queue name not be displayed.

5
```
$ PRINT/REMOTE BOSTON::WORK$:[SMITH.MEMO]JUNE10.MEM
  Job JUNE10 (queue SYS$PRINT, entry 476) started on LPA0
```

The PRINT command in this example, which is entered on a node other than BOSTON, queues the file JUNE10.MEM that resides on the BOSTON node. The file is entered on the printer queue at node BOSTON.

6
```
$ COPY REPORT.MEM BOSTON::
$ PRINT/REMOTE BOSTON::REPORT.MEM
  Job REPORT (queue SYS$PRINT, entry 342) started on LPA0
```

In this example, the two commands are entered at a node other than BOSTON. The COPY command copies the file REPORT.MEM from the current node to the BOSTON node. The PRINT command queues the file REPORT.MEM located on the BOSTON node for printing at the BOSTON node.

7
```
$ PRINT/HOLD   MASTER.DOC
  Job MASTER (queue SYS$PRINT, entry 540) holding
    .
    .
    .
$ SET QUEUE/ENTRY=540/RELEASE
```

The PRINT command in this example queues a copy of the file MASTER.DOC to the default printer in a hold status. Later, the SET QUEUE/ENTRY command releases the hold status on the file and makes it available for printing.

PURGE

Deletes all but the highest-numbered versions of the specified files.

FORMAT

PURGE *[file-spec[,...]]*

PARAMETER

file-spec[,...]

Specifies one or more files to be purged. If you specify two or more files, separate them with either commas or plus signs. Wildcard characters are allowed in the directory, file name, and file type fields; however, no version number can be specified. As a default, the PURGE command purges all files in the current directory. There are no file name or file type defaults with the PURGE command.

DESCRIPTION

The PURGE command deletes earlier versions of files. The PURGE command never deletes all versions of any file. By default, the PURGE command keeps only the highest version of a file. If you do not include a file specification with the PURGE command, all files in the current directory are affected by the purge.

QUALIFIERS

/BACKUP

Modifies the time value specified with the /BEFORE or /SINCE qualifier. /BACKUP selects files according to the dates of their most recent backups. This qualifier is incompatible with the other qualifiers that also allow you to select files according to time attributes: /CREATED, /EXPIRED, and /MODIFIED. If you specify none of these four time qualifiers, the default is /CREATED.

/BEFORE[=time]

Selects only those files dated prior to the specified time. You can specify time as an absolute time, as a combination of absolute and delta times, or as one of the following keywords: TODAY (default), TOMORROW, or YESTERDAY. Specify one of the following qualifiers with /BEFORE to indicate the time attribute to be used as the basis for selection: /BACKUP, /CREATED (default), /EXPIRED, or /MODIFIED.

See Section 1.4 of the *VMS DCL Concepts Manual* for complete information on specifying time values.

/BY_OWNER[=uic]

Selects only those files whose owner user identification code (UIC) matches the specified owner UIC. The default UIC is that of the current process.

Specify the UIC using standard UIC format as described in Section 8.1 of the *VMS DCL Concepts Manual.*

/CONFIRM
/NOCONFIRM (default)

Controls whether a request is issued before each PURGE operation to confirm that the operation should be performed on that file. The following responses are valid:

YES	NO	QUIT
TRUE	FALSE	CTRL/Z
1	0	ALL
	RET	

You can use any combination of uppercase and lowercase letters for word responses. Word responses can be abbreviated to one or more letters (for example, T, TR, or TRU for TRUE), but these abbreviations must be unique. Affirmative answers are YES, TRUE, and 1. Negative answers are NO, FALSE, 0, and <RET>. QUIT or CTRL/Z indicates that you want to stop processing the command at that point. When you respond with ALL, the command continues to process, but no further prompts are given. If you type a response other than one of those in the list, DCL issues an error message and redisplays the prompt.

/CREATED (default)

Modifies the time value specified with the /BEFORE or /SINCE qualifier. /CREATED selects files based on their dates of creation. This qualifier is incompatible with the other qualifiers that also allow you to select files according to time attributes: /BACKUP, /EXPIRED, and /MODIFIED. If you specify none of these four time qualifiers, the default is /CREATED.

/ERASE
/NOERASE (default)

Erases the specified files from the disk so that the purged data no longer exists physically on the deallocated disk blocks.

When you delete a file, the area in which the file was stored is returned to the system for future use. The data that was stored in that location still exists in the system until new data is written over it. When the /ERASE qualifier is specified, the storage location is overwritten with a system specified pattern so that the data no longer exists.

/EXCLUDE=(file-spec[,...])

Excludes the specified files from the PURGE operation. You can include a directory but not a device in the file specification. Wildcard characters are allowed in the file specification. However, you cannot use relative version numbers to exclude a specific version. If you provide only one file specification, you can omit the parentheses.

/EXPIRED

Modifies the time value specified with the /BEFORE or /SINCE qualifier. /EXPIRED selects files according to their expiration dates. (The expiration date is set with the SET FILE/EXPIRATION_DATE command.) The /EXPIRED qualifier is incompatible with the other qualifiers that also allow you to select files according to time attributes: /BACKUP, /CREATED, and /MODIFIED. If you specify none of these four time qualifiers, the default is /CREATED.

/KEEP=number-of-versions

Specifies the maximum number of versions of the specified files to be retained in the directory. If you do not include the /KEEP qualifier, all but the highest-numbered version of the specified files are deleted from the directory.

/LOG
/NOLOG (default)

Controls whether file specifications are displayed as the files are deleted.

/MODIFIED

Modifies the time value specified with the /BEFORE or /SINCE qualifier. /MODIFIED selects files according to the dates on which they were last modified. This qualifier is incompatible with the other qualifiers that also allow you to select files according to time attributes: /BACKUP, /CREATED, and /EXPIRED. If you specify none of these four time modifiers, the default is /CREATED.

/SINCE[=time]

Selects only those files dated after the specified time. You can specify time as an absolute time, a combination of absolute and delta times, or as one of the following keywords: TODAY (default), TOMORROW, or YESTERDAY. Specify one of the following qualifiers with /BEFORE to indicate the time attribute to be used as the basis for selection: /BACKUP, /CREATED (default), /EXPIRED, or /MODIFIED.

See Section 1.4 of the *VMS DCL Concepts Manual* for complete information on specifying time values.

EXAMPLES

1
```
$ PURGE
```

The PURGE command in this example deletes all but the highest-numbered version of all files in the default directory.

2
```
$ PURGE *.COM
```

The PURGE command in this example deletes all but the highest-numbered version of each file with a file type of COM.

3
```
$ PURGE/KEEP=3  [WILDER.JOB308]ACCOUNT.COB
```

The PURGE command in this example deletes all but the three highest-numbered versions of the file ACCOUNT.COB in the subdirectory [WILDER.JOB308].

4
```
$ PURGE/ERASE/SINCE=YESTERDAY [.MEMOS]
```

The PURGE command in this example purges all files in the MEMOS subdirectory that have been created or modified since yesterday and erases the storage locations so that the purged data no longer exists.

5
```
$ PURGE [MAL.TESTFILES]/LOG
%PURGE-I-FILPURG, DISK1:[MAL.TESTFILES]AVE.OBJ;1 deleted (3 blocks)
%PURGE-I-FILPURG, DISK1:[MAL.TESTFILES]BACK.OBJ;2 deleted (5 blocks)
%PURGE-I-TOTAL, 2 files deleted (8 blocks)
```

The PURGE command in this example purges all files cataloged in the subdirectory named [MAL.TESTFILES]. The /LOG qualifier requests the PURGE command to display the specification of each file it has deleted as well as the total number of files that have been deleted.

6
```
$ PURGE/KEEP=2 TAMPA::DISK1:[EXAMPLE]*.LIS
```

The PURGE command in this example deletes all but the two highest-numbered versions of each file with the file type LIS in the directory EXAMPLE on remote node TAMPA.

READ

Reads a single record from a specified input file and assigns the record's contents to a specified symbol name.

FORMAT

READ *logical-name[:] symbol-name*

PARAMETERS

logical-name[:]

Specifies the logical name of the input file from which a record is to be read. Use the logical name assigned by the OPEN command when the file was opened. (The OPEN command assigns a logical name to a file and places the name in the process logical name table.)

In addition, you can specify the process-permanent files identified by the logical names SYS$INPUT, SYS$OUTPUT, SYS$ERROR, and SYS$COMMAND.

symbol-name

Specifies the name of a symbol to be equated to the contents of the record. The name must be 1 through 255 alphanumeric characters and must start with an alphabetic letter, underscore, or dollar sign.

When you specify a symbol name for the READ command, the command interpreter places the symbol name in the local symbol table for the current command level. If the symbol has already been defined, the READ command redefines it to the new value being read.

DESCRIPTION

The READ command can read data from sequential, relative, or indexed sequential files. After each record is read from the specified file, the READ command positions the record pointer at the next record in the file. However, if you are reading an ISAM file, you can use the /INDEX and /KEY qualifiers to read records randomly.

The maximum size of any record that can be read in a single READ command is 2048 bytes.

To read a file, the file must be opened using the /READ qualifier with the OPEN command. The process-permanent files identified by the logical names SYS$INPUT, SYS$OUTPUT, SYS$ERROR, and SYS$COMMAND do not have to be explicitly opened to be read.

If the READ command is executed interactively and the logical name is specified as one of the process-permanent files, SYS$INPUT, SYS$OUTPUT, SYS$COMMAND, or SYS$ERROR, the command interpreter prompts for input data. The READ command accepts data exactly as you enter it. The READ command does not convert characters to uppercase, remove extra spaces and tabs, or remove quotation marks. Also, the READ command does not perform symbol substitution. See the /PROMPT qualifier for more information on issuing prompts with the READ command.

QUALIFIERS

/DELETE

Deletes a record from an ISAM file after it has been read. An ISAM file must be opened with the /READ and /WRITE qualifiers in order to use READ /DELETE.

/END_OF_FILE=label

Transfers control to the location specified by the label keyword (in the current command procedure) when the end of the file is reached. When the last record in the file is read, the VAX Record Management Services (VAX RMS) return an error condition indicating the end-of-file. If the /END_OF_FILE qualifier is specified, the command interpreter transfers control to the command line at the specified label.

If /END_OF_FILE is not specified, control is given to the error label specified with the /ERROR qualifier when the end-of-file is reached. If neither /ERROR nor /END_OF_FILE is specified, then the current ON condition action is taken.

/ERROR=label

Transfers control to the location specified by the label keyword in the current command procedure) when a read error occurs. If no error routine is specified and an error occurs during the reading of the file, the current ON condition action is taken.

Overrides any ON condition action specified.

If an error occurs and the target label is successfully given control, the reserved global symbol $STATUS retains the error code.

/INDEX=n

Specifies the index (n) to be used to look up keys when reading an ISAM file.

The default value is 0, the primary index.

/KEY=string

Reads a record with the key that matches the specified character string. Binary and integer keys are not allowed. This qualifier, when used together with /INDEX, allows you random access to ISAM files.

Key matches are made by comparing the characters in the /KEY string to characters in the record key.

To read records at random in an ISAM file, you must specify the /KEY qualifier. Once a record is read randomly, all subsequent reads without the /KEY qualifier access records in the ISAM file sequentially.

/MATCH=option

Specifies the ISAM key match algorithm to be used when searching for matching keys. Specify one of the following options:

EQ	Select keys equal to the match value (default)
GE	Select keys greater than or equal to the match value
GT	Select keys greater than the specified key

If you are reading ISAM files and you do not use the /MATCH qualifier, the default is /MATCH=EQ.

/NOLOCK

Specifies that the record to be read not be locked and enables a record to be read that has been locked by other accessors.

By default, records are locked as they are read and unlocked on the next I/O operation on the file.

/PROMPT=string

Specifies an alternate prompt string to be displayed when reading from the terminal. The default prompt string is DATA:.

/TIME_OUT=n
/NOTIME_OUT (default)

Specifies the number of seconds after which the READ command is terminated if no input is received. If you enter the /TIME_OUT qualifier, you must specify a value from 0 through 255.

If you enter both the /ERROR=label and /TIME_OUT qualifiers, and the time limit expires, the error branch is taken.

EXAMPLES

1

```
$ OPEN IN NAMES.DAT
$ LOOP:
$ READ/END_OF_FILE=ENDIT IN NAME
    .
    .
    .
$ GOTO LOOP
$ ENDIT:
$ CLOSE IN
```

The OPEN command opens the file NAMES.DAT for input and assigns it the logical name of IN. The READ command reads records from the file IN and places the contents into the symbol NAME. The READ command specifies the label ENDIT to receive control when the last record in the file has been read. The procedure loops until all records in the file have been processed.

2

```
$ READ/ERROR=READERR/END_OF_FILE=OKAY  MSGFILE  CODE
    .
    .
    .
$ READERR:
$ CLOSE MSGFILE
    .
    .
    .
$ OKAY:
$ CLOSE MSGFILE
$ EXIT
```

The READ command reads records from the file MSGFILE and places the contents into the symbol CODE. The READ command also uses the /ERROR and /END_OF_FILE qualifiers to specify labels to receive control at the end-of-file and on error conditions. At the end-of-file, control is transferred to the label OKAY. On other read errors, control is transferred to the READERR label.

3
```
$ READ SYS$COMMAND  DATA_LINE
$ WRITE  OUTPUT_FILE  DATA_LINE
    .
    .
    .
```

The READ command requests data from the current SYS$COMMAND device. If the command procedure containing these lines is executed interactively, the command issues a prompt to the terminal, accepts a line of data, and equates the data entered to the symbol name DATA_LINE.

Then the WRITE command writes the value of the symbol DATA_LINE to the file identified by the logical name OUTPUT_FILE.

4
```
$ OPEN/READ INPUT_FILE TRNTO::INVENTORY.DAT
$ OPEN/APPEND OUTPUT_FILE RECEIVE.DAT
$ READ INPUT_FILE DATA_LINE
$ WRITE OUTPUT_FILE DATA_LINE
```

The OPEN/READ command opens the file INVENTORY.DAT at the remote node TRNTO for reading and assigns it the logical name INPUT_FILE. The OPEN/APPEND command opens the file RECEIVE.DAT in the current default directory. The READ command requests data from the file INVENTORY.DAT at the remote node TRNTO. The WRITE command writes the value of the symbol DATA_LINE to the end of the local file RECEIVE.DAT.

RECALL

Displays previously entered commands on the screen for subsequent execution.

FORMAT

RECALL *[command-specifier]*

PARAMETER

command-specifier

Specifies the number or the first several characters of the command you want to recall.

The specified characters should be unique. If they are not unique, the RECALL command displays the most recently entered command line that matches those characters. For example, suppose you enter a SHOW STATUS command and later enter a SHOW TIME command. If you then type RECALL S, the SHOW TIME command is recalled. You must type RECALL SHOW S to recall the SHOW STATUS command. The number of the command can be from 1 to 20 (where 1 is the last command entered).

The RECALL command itself is never assigned a number. If no command specifier is entered, the RECALL command recalls the most recently entered command. You can use the /ALL qualifier to have the system display all the commands in the RECALL buffer, along with their command numbers, to verify the number of the command you want to recall.

DESCRIPTION

When you enter commands to the system, they are stored in a RECALL buffer for later use with the RECALL command. Input to the INQUIRE command in command procedures is also placed in the RECALL buffer. The RECALL command itself is never stored in the RECALL buffer.

The RECALL buffer can hold up to 20 commands or 1025 characters. You can use continuation characters with the commands to be recalled, but only 255 characters can be read at a time.

When you use the RECALL command, the system displays the command but does not process it. If you want it processed as it appears, press RETURN. You can use the command line editing facility to make minor changes in the command line and then press RETURN to process the revised version of the command.

QUALIFIERS

/ALL

Displays all the commands (and their numbers) available for recall. Remember that the RECALL command is never stored in the RECALL buffer.

/ERASE

Erases the contents of the recall buffer.

EXAMPLES

1

```
$ RECALL T
```

The RECALL command in this example recalls the last command entered that begins with the letter "T".

2

```
$ SHOW DEFAULT
  DISK3:[SMITH]
$ DIRECTORY SEPT*
  %DIRECT-W-NOFILES, no files found
$ SET DEFAULT [SMITH.LETTERS]
$ RECALL/ALL
  1 SET DEFAULT [SMITH.LETTERS]
  2 DIRECTORY SEPT*
  3 SHOW DEFAULT
$ RECALL 2
$ DIRECTORY SEPT* <RETURN>
  %DIRECT-W-NOFILES, no files found
$ RECALL 2
$ SET DEFAULT [SMITH.LETTERS]
  <DELETE> <DELETE> <DELETE> <DELETE> <DELETE> <DELETE> <DELETE> <DELETE>
$ SET DEFAULT [SMITH.MEMOS] <RETURN>
$ RECALL 2
$ DIRECTORY SEPT* <RETURN>
```

This example starts with a SHOW DEFAULT and a DIRECTORY command. Not finding the file you want, enter the SET DEFAULT command to move to the LETTERS subdirectory. Next use the RECALL/ALL command to see the list of commands you have entered. Enter the RECALL 2 command to repeat the DIRECTORY command in the LETTERS subdirectory. Because you still have not found the file you want, enter the RECALL 2 command again to recall the SET DEFAULT command. (With the entry of the latest DIRECTORY command, SET DEFAULT becomes command 2 in the RECALL list.) Using the DELETE key, edit the command line so that the system sets the default to the MEMOS subdirectory. Finally, recall the DIRECTORY command to try once more to find the file. Press RETURN to process the recalled command.

RENAME

Changes all or part of a file specification of an existing disk file or disk directory.

FORMAT

RENAME *input-file-spec[,...] output-file-spec*

PARAMETERS

input-file-spec[,...]

Specifies the name of one or more files whose specifications are to be changed. Wildcard characters are allowed in the directory specification, file name, file type, or version number fields of the file specification. When wildcard characters are used, all files whose specifications satisfy the wildcard fields are renamed.

output-file-spec

Provides the new file specification to be applied to the input file. The RENAME command uses the device, directory, file name, and file type of the input file as defaults for fields in the output file that are either not specified, or indicated by a wildcard character. Wildcard characters in corresponding fields of the input and output file specification result in multiple rename operations. The RENAME command supplies output file version numbers in the following ways:

1. If the output file specification contains an explicit version number, that version number is used.
2. If the output file specification contains a wildcard as the version number, the version number of the input file is used.
3. If the input file specification contains a wildcard as the version number, the version number of each input file names a corresponding output file.
4. If no file exists with the same file name and type as the output file, version 1 is used.
5. If a file already exists with the same file name and type as the output file, the next higher version number is used (unless the /NONEWVERSION qualifier is specified).

DESCRIPTION

The RENAME command changes the directory name, file name, file type, or version number of a file. The node and disk designation for the input file specification must be the same as that for the output file specification. In addition, you must have delete access privileges to a file to rename the file. You cannot rename a file across a network.

QUALIFIERS

/BACKUP

Modifies the time value specified with the /BEFORE or /SINCE qualifier. /BACKUP selects files according to the dates of their most recent backups. This qualifier is incompatible with the other qualifiers that also allow you to select files according to time attributes: /CREATED, /EXPIRED, and

/MODIFIED. If you specify none of these four time qualifiers, the default is /CREATED.

/BEFORE[=time]

Selects only those files dated prior to the specified time. You can specify time as an absolute time, as a combination of absolute and delta times, or as one of the following keywords: TODAY (default), TOMORROW, or YESTERDAY. Specify one of the following qualifiers with /BEFORE to indicate the time attribute to be used as the basis for selection: /BACKUP, /CREATED (default), /EXPIRED, or /MODIFIED.

See Section 1.4 of the *VMS DCL Concepts Manual* for complete information on specifying time values.

/BY_OWNER[=uic]

Selects only those files whose owner user identification code (UIC) matches the specified owner UIC. The default UIC is that of the current process.

Specify the UIC using standard UIC format as described in Section 8.1 of the *VMS DCL Concepts Manual*.

/CONFIRM
/NOCONFIRM (default)

Controls whether a request is issued before each RENAME operation to confirm that the operation should be performed on that file. The following responses are valid:

YES	NO	QUIT
TRUE	FALSE	CTRL/Z
1	0	ALL
	RET	

You can use any combination of uppercase and lowercase letters for word responses. Word responses can be abbreviated to one or more letters (for example, T, TR, or TRU for TRUE), but these abbreviations must be unique. Affirmative answers are YES, TRUE, and 1. Negative answers are NO, FALSE, 0, and <RET>. QUIT or CTRL/Z indicates that you want to stop processing the command at that point. When you respond with ALL, the command continues to process, but no further prompts are given. If you type a response other than one of those in the list, DCL issues an error message and redisplays the prompt.

/CREATED (default)

Modifies the time value specified with the /BEFORE or /SINCE qualifier. /CREATED selects files based on their dates of creation. This qualifier is incompatible with the other qualifiers that also allow you to select files according to time attributes: /BACKUP, /EXPIRED, and /MODIFIED. If you specify none of these four time qualifiers, the default is /CREATED.

/EXCLUDE=(file-spec[,...])

Excludes the specified files from the RENAME operation. You can include a directory but not a device in the file specification. Wildcard characters are allowed in the file specification. However, you cannot use relative version numbers to exclude a specific version. If you provide only one file specification, you can omit the parentheses.

/EXPIRED

Modifies the time value specified with the /BEFORE or /SINCE qualifier. /EXPIRED selects files according to their expiration dates. This qualifier is incompatible with the other qualifiers that also allow you to select files according to time attributes: /BACKUP, /CREATED, and /MODIFIED. If you specify none of these four time qualifiers, the default is /CREATED.

/LOG
/NOLOG (default)

Displays the file specification of each file as it is renamed.

/MODIFIED

Modifies the time value specified with the /BEFORE or /SINCE qualifier. /MODIFIED selects files according to the dates on which they were last modified. This qualifier is incompatible with the other qualifiers that also allow you to select files according to time attributes: /BACKUP, /CREATED, and /EXPIRED. If you specify none of these four time modifiers, the default is /CREATED.

/NEW_VERSION (default)
/NONEW_VERSION

Assigns a new version number if an output file specification is the same as that of an existing file. The /NONEW_VERSION qualifier displays an error message if an output file specification is the same as that of an existing file. A wildcard appearing in the version field of an input or output file overrides these qualifiers.

/SINCE[=time]

Selects the RENAME operation only for those files dated after the specified time. You can specify time as an absolute time, a combination of absolute and delta times, or as one of the following keywords: TODAY (default), TOMORROW, or YESTERDAY. Specify one of the following qualifiers with /BEFORE to indicate the time attribute to be used as the basis for selection: /BACKUP, /CREATED (default), /EXPIRED, or /MODIFIED.

See Section 1.4 of the *VMS DCL Concepts Manual* for complete information on specifying time values.

EXAMPLES

1
```
$ RENAME  AVERAGE.OBJ  OLDAVERAGE
```

The RENAME command in this example renames the highest existing version of the file AVERAGE.OBJ to OLDAVERAGE.OBJ. If no file named OLDAVERAGE.OBJ currently exists, the new file is assigned a version number of 1.

2
```
$ RENAME/NONEW_VERSION  SCANLINE.OBJ;2  BACKUP.OBJ
```

The RENAME command in this example renames the file SCANLINE.OBJ;2 to BACKUP.OBJ;2. The /NONEW_VERSION qualifier ensures that, if BACKUP.OBJ;2 already exists, the RENAME command does not rename the file, but instead reports the error.

```
3  $ RENAME  *.TXT;*   *.OLD;*
```

The RENAME command in this example renames all versions of all files with the file type TXT to have the file type OLD. The file names and version numbers are not changed.

```
4  $ RENAME WATER.TXT [.MEMOS]
```

The RENAME command in this example changes the directory name of WATER.TXT from your default directory to the MEMOS subdirectory. (The RENAME command moves the file to another directory.)

```
5  $ RENAME  [MALCOLM.TESTFILES]SAVE.DAT  []TEST
```

The RENAME command in this example renames the file SAVE.DAT in the directory MALCOLM.TESTFILES to TEST.DAT. The new file is moved to the current default directory.

```
6  $ RENAME/LOG
   $_From:      DATA.*,INFO.*
   $_To:        NEW
   %RENAME-I-RENAMED, _DISKO:[SYSTEM]DATA.AAA;1 renamed to _DISKO:[SYSTEM]NEW.AAA;1
   %RENAME-I-RENAMED, _DISKO:[SYSTEM]DATA.BBB;1 renamed to _DISKO:[SYSTEM]NEW.BBB;1
   %RENAME-I-RENAMED, _DISKO:[SYSTEM]DATA.CCC;1 renamed to _DISKO:[SYSTEM]NEW.CCC;1
   %RENAME-I-RENAMED, _DISKO:[SYSTEM]INFO.001;1 renamed to _DISKO:[SYSTEM]NEW.001;1
   %RENAME-I-RENAMED, _DISKO:[SYSTEM]INFO.002;1 renamed to _DISKO:[SYSTEM]NEW.002;1
   %RENAME-I-RENAMED, _DISKO:[SYSTEM]INFO.003;1 renamed to _DISKO:[SYSTEM]NEW.003;1
   $
```

In this example, three files exist with the file name of DATA, and three files have the file name of INFO. This RENAME command illustrates wildcard characters in the input file names and the use of temporary default file types and version numbers on the output files. The result is the renaming of all six files as displayed by the /LOG qualifier.

```
7  $ RENAME NODE1::DISK2:[SMITH]ASSEMSHT.EXE NODE1::DISK3:[JONES]ASSEMBLYSHEET.EXE
```

The RENAME command in this example renames the file ASSEMSHT.EXE in the SMITH directory on remote node NODE1 and disk DISK3 to ASSEMBLYSHEET.EXE in the JONES directory on the same remote node and disk. You can rename a file on another node and disk only if the new file resides on that same node and disk.

REPLY

Broadcasts a message to a terminal or terminals.

See the qualifier descriptions for restrictions.

FORMAT

REPLY *["message-text"]*

PARAMETER

message-text

Specifies the text of the message. The text must be 1 to 128 characters. Enclose the text in quotation marks ("") if it contains spaces, special characters, or lowercase characters.

DESCRIPTION

All users with OPER privilege can use the REPLY command to communicate with system users. The REPLY command does the following:

- Displays messages at users' terminals
- Responds to user requests
- Responds to magnetic tape file system requests
- Enables and disables operator status on a terminal (if the Operator Communication Facility (OPCOM) is running)
- Closes the operator's log file and opens a new one (if the Operator Communication Facility (OPCOM) is running)

You must always use one or more qualifiers with the REPLY command in order for it to be meaningful. If you use the REPLY command without using any qualifiers, an error message is returned. When you use the REPLY command for any purpose other than displaying messages at users' terminals, you must also use the /ENABLE=keyword qualifier. See the description of the /ENABLE qualifier to determine the appropriate keyword (or keywords) for your purpose.

Displaying Messages at Users' Terminals

To contact one or more system users, the operator enters one of the following REPLY commands:

- REPLY/ALL "message text"
- REPLY/TERMINAL=(terminal-name[,...]) "message text"
- REPLY/USERNAME[=(username[,...])] "message text"

The /ALL qualifier sends a message to all terminals that are online and connected to the VMS system or VAXcluster. Generally, when an important message is to be broadcast, such as information about a system shutdown, you should use the /ALL qualifier.

The /TERMINAL qualifier sends a message to one or more specific terminals on the system or VAXcluster.

The /USERNAME qualifier sends a message to terminals at which one or more system or VAXcluster users are logged in.

Note that the /TO qualifier is *not* used under these three circumstances, because the operator is not replying to a specific request from either the file system or a user.

To broadcast to a terminal other than your own, you must have OPER privilege. The REPLY command is not complete until all terminals you are broadcasting to have received the message.

Responding to User Requests

When a user enters the REQUEST/REPLY command, the process associated with the requesting user's terminal is put in a wait state until the operator responds using one of the following REPLY commands:

- REPLY/ABORT=identification-number "message-text"
- REPLY/PENDING=identification-number "message-text"
- REPLY/TO=identification-number "message-text"

The /ABORT qualifier indicates that the user's request has been canceled.

The /PENDING qualifier sends a message to the user and keeps the user's process in a wait state until the request can be fulfilled or aborted.

The /TO qualifier indicates that the user's request has been fulfilled.

When a user enters the REQUEST/REPLY command, the message is displayed at the system console terminal. For example:

```
%OPCOM, 31-DEC-1988 09:49:24.47, request 3, from user SYSTEM
_TTB6:, This is a sample request
```

The user cannot enter any further commands until the operator responds using the /ABORT or /TO qualifier, or until the user aborts the request. If the operator does not respond and the user does not abort the request, the request is repeated at five-minute intervals on the operator's terminal until the operator replies.

The REPLY command is an essential part of the procedures that operators must use in order for users to gain access to tape and disk volumes.

Responding to File System Requests

When a multivolume tape volume reaches the end-of-tape mark, the magnetic tape file system suspends processing and sends a message to the operator to mount the next tape. The operator responds using one of the following REPLY commands:

- REPLY/TO=identification-number "label"
- REPLY/INITIALIZE_TAPE=identification-number "label"
- REPLY/BLANK_TAPE=identification-number "label"
- REPLY/ABORT=identification-number

The /TO qualifier indicates that the file system request has been fulfilled. When the request from the magnetic tape file system specifies a volume label, the operator mounts the specified tape and enters the REPLY/TO command. However, if the file system requests a new volume, the operator can mount a scratch tape and enter the REPLY/INITIALIZE_TAPE or REPLY/BLANK_TAPE command with the message "label". The double quotation marks are required syntax.

If the request is "REMOUNT" or "MOUNT NEW", the label is required in the message text. If the request is "MOUNT", no label is needed.

The /ABORT qualifier indicates that the file system request has been canceled.

Enabling and Disabling Operator Status on a Terminal

Any terminal connected to the VMS operating system can be established as an operator's terminal if the Operator Communication Facility (OPCOM) is running. When an operator who is logged in to an account with operator privilege enters the REPLY/ENABLE command at the designated terminal, that terminal can be used to respond to user requests and to monitor device status. Such a terminal retains operator status until it is specifically disabled, or until the end of the current interactive session, if it was established as a temporary operator's terminal (see /TEMPORARY).

Operator messages are printed on the system console terminal unless that terminal is explicitly disabled as an operator's terminal.

When the operator enters the REPLY/ENABLE command, the Operator Communication Facility (OPCOM) confirms that the terminal has been enabled. For example:

```
$ REPLY/ENABLE

%OPCOM, 31-DEC-1988 10:22:19.75, operator status for operator OPA0
CENTRAL, PRINTER, TAPES, DISKS, DEVICES, CARDS, NETWORK, CLUSTER,
OPER1, OPER2, OPER3, OPER4, OPER5, OPER6, OPER7, OPER8, OPER9, OPER10,
OPER11, OPER12
```

When the operator enters the REPLY/DISABLE command, OPCOM uses the following message to confirm that the terminal is no longer an operator terminal:

```
%OPCOM, 31-DEC-1988 10:03:23.48, operator disabled, operator OPA0
```

To grant specific operator status on a particular terminal, the operator includes one or more keywords after the /ENABLE qualifier. For example, to establish a terminal as an operator terminal that can receive messages pertaining to mounting and dismounting tapes and disks, the operator enters the following:

```
$ REPLY/ENABLE=(DISKS,TAPES)

%OPCOM, 31-DEC-1988 10:04:00.18, operator enabled, operator OPA0
$
%OPCOM, 31-DEC-1988 10:04:00.47, operator status for operator OPA0
TAPES, DISKS
```

OPCOM confirms that the terminal has operator status for tape and disk messages.

To discontinue specific operator status, the operator includes one or more keywords after the /DISABLE qualifier. For example, to inhibit an operator terminal from receiving messages pertaining to mounting and dismounting disks, the operator enters the following command:

```
$ REPLY/DISABLE=DISKS

%OPCOM, 31-DEC-1988 10:04:30.83, operator status for operator OPA0
TAPES
```

Note that OPCOM lists the specific operator status still assigned to the terminal.

When an operator disables operator status on all terminals, including the system console terminal, OPCOM records all subsequent messages in the operator log file, except user requests and messages requiring an operator reply.

```
%OPCOM-S-OPRNOTIF, operator notified, waiting. 10:06:03.25
%OPCOM-S-OPREPLY, %OPCOM 31-DEC-1988 10:06:03:25, no operator coverage
```

To determine the operator status and obtain a list of pending requests for a particular terminal, the operator enters the following command:

```
$ REPLY/STATUS
```

This command also shows all outstanding requests for this operator.

Closing the Operator's Log File and Opening a New One

To close the current operator's log file and open a new one, the operator enters the REPLY/LOG command. If the Operator Communication Facility (OPCOM) is running, all subsequent messages are recorded in the new log file. To close the current log file without opening a new one, the operator enters the REPLY/NOLOG command. All subsequent messages are not recorded until the operator enters the REPLY/LOG command.

QUALIFIERS

/ABORT=identification-number

Sends a message to the user or magnetic tape file system corresponding to the unique identification number and cancels the request.

/ALL

Requires OPER privilege.

Broadcasts a message to all terminals that are attached to the system or VAXcluster. These terminal must be turned on and have broadcast-message reception enabled. Incompatible with /USERNAME and /TERMINAL.

/BELL

Rings a bell at the terminal receiving a message when entered with the /ALL, /TERMINAL, or /USER qualifiers; two bells when entered with /URGENT; and three bells when entered with /SHUTDOWN.

/BLANK_TAPE=identification-number

Requires VOLPRO privilege.

Sends a message to the magnetic tape file system indicated by the identification number to override the checking of volume label information. The volume label must be specified in the message text parameter. The current terminal must be enabled as an operator terminal for TAPES.

/DISABLE[=(keyword[,...])]

Requires OPER privilege. Requires OPER and SECURITY privileges for security messages.

If the Operator Communication Facility (OPCOM) is running, restores to normal (that is, nonoperator) status the terminal at which the command is entered. The /DISABLE qualifier cannot be entered from a batch job. To restrict the types of messages displayed on an operator's terminal, specify one of the following keywords:

CARDS	Inhibits messages sent to the card readers
CENTRAL	Inhibits messages sent to the central system operator
CLUSTER	Inhibits messages from the connection manager pertaining to cluster state changes
DEVICES	Inhibits messages pertaining to mounting disks
DISKS	Inhibits messages pertaining to mounting and dismounting disk volumes
NETWORK	Inhibits messages pertaining to networks; the keyword CENTRAL must also be specified to inhibit network messages
OPER1 through OPER12	Inhibits messages sent to operators identified as OPER1 through OPER12
PRINTER	Inhibits messages pertaining to print requests
SECURITY	Inhibits messages pertaining to security events; requires SECURITY privilege.
TAPES	Inhibits messages pertaining to mounting and dismounting tape volumes

When an operator logs out, the operator terminal is automatically disabled.

/ENABLE[=(keyword[,...])]

Requires OPER privilege. Requires OPER and SECURITY privileges for security messages.

If the Operator Communication Facility (OPCOM) is running, designates as an operator's terminal the terminal at which the REPLY command is entered. Cannot be entered from a batch job.

CARDS	Displays messages sent to the card readers
CENTRAL	Displays messages sent to the central system operator
CLUSTER	Displays messages from the connection manager pertaining to cluster state changes
DEVICES	Displays messages pertaining to mounting disks
DISKS	Displays messages pertaining to mounting and dismounting disk volumes
NETWORK	Displays messages pertaining to networks; the keyword CENTRAL must also be specified to inhibit network messages

OPER1 through OPER12	Displays messages sent to operators identified as OPER1 through OPER12
PRINTER	Displays messages pertaining to print requests
SECURITY	Allows messages pertaining to security events; requires SECURITY privilege
TAPES	Allows messages pertaining to mounting and dismounting tape volumes

/INITIALIZE_TAPE=identification-number

Sends a message to the magnetic tape file system indicated by the identification number to initialize a magnetic tape volume. This qualifier can be used whenever the file system requests the mounting of a new volume. The system performs normal protection and expiration checks before initializing the volume. The current terminal must be enabled as an operator terminal for TAPES.

/LOG
/NOLOG

Requires OPER privilege.

If the Operator Communication Facility (OPCOM) is running, closes the current operator's log file and opens a new one. (The /NOLOG qualifier closes the current log file, but does not open a new log file.) The current terminal must be enabled as an operator terminal. The operator can then examine the contents of the previous log file.

/NODE[=(node-name[,...])]

Sends a message to the local VAXcluster node only. The optional parameter list allows you to specify which nodes will receive the message. Default sends messages to all cluster nodes.

/NOTIFY (default)
/NONOTIFY

Sends a message describing success back to the originating terminal.

/PENDING=identification-number

Requires OPER privilege.

Sends a message to the user specified by the identification number and prevents the user from entering other commands until the operator fulfills or aborts the request. The current terminal must be enabled as an operator terminal.

/SHUTDOWN

Sends a message beginning "SHUTDOWN..."; if used with /BELL, rings three bells at terminals receiving the message.

/STATUS

Requires OPER privilege.

Reports the current operator status and all outstanding user requests for the terminal from which this command was entered. The current terminal must be enabled as an operator terminal.

/TEMPORARY
Designates the terminal at which the command is entered to be an operator's terminal for the current interactive session only. This qualifier is meaningful only when used with the /ENABLE qualifier.

/TERMINAL=(terminal-name[,...])
Requires OPER privilege.

Broadcasts the message to specified terminals, where the terminal-name keyword is the device name of the terminal. Incompatible with /ALL and /USERNAME.

/TO=identification-number
Requires OPER privilege.

Sends a message to the user or file system specified by the identification number and completes the request. The current terminal must be enabled as an operator terminal.

Note that you can also use a variation of REPLY/TO in response to a MOUNT/ASSIST command where you redirect the mount operation to another device. Whenever you must substitute a device, load the user's volume on the alternate device and ready the device before entering the REPLY command. Use the following syntax:

```
REPLY/TO=identification-number "SUBSTITUTE   device-name"
```

You can abbreviate the word SUBSTITUTE to **S** and use upper or lowercase characters. After a space, use the remainder of the message-text space to name the substituted device.

/URGENT
Sends a message beginning "URGENT..."; if used with the /BELL qualifier, rings two bells at terminals receiving the message.

/USERNAME[=(username[,...])]
Requires OPER privilege.

Broadcasts a message to all terminals at which users are logged in to the system (or VAXcluster), or only to the terminals of the specified users. Overrides any NOBROADCAST settings at users' terminals.

/WAIT
Sends a message synchronously and then waits. The default is to send a message to OPCOM, which does the actual I/O. On a VAXcluster, the message is sent to the local node.

EXAMPLES

1
```
$ REPLY/ALL/BELL "SYSTEM GOING DOWN FOR BACK-UP. PLEASE LOG OFF."
```

The REPLY command in this example broadcasts a message to all terminals on the system. When the message appears at the user's terminal, it is prefixed with terminal name, the user name of the sender, and (when DECnet–VAX is installed) the node name. The bell sounds at the terminal as the message is displayed.

2

```
$ REPLY/ENABLE=DISKS
%OPCOM, 31-DEC-1988, 10:17:09.02, operator enabled, operator OPA0
$
%OPCOM, 31-DEC-1988 10:17:10.30, operator status for operator OPA0
DISKS
```

The REPLY/ENABLE command in this example designates the terminal OPA0 as an operator terminal that can receive messages pertaining to mounting and dismounting disks. The OPCOM message confirms that terminal OPA0 is established as an operator's terminal.

3

```
%OPCOM, 31-DEC-1988 10:19:33.21, request 5, from user SYSTEM
OPA0, Please mount OPGUIDE on DBA3:
$ REPLY/PENDING=5 "YOU'LL HAVE TO WAIT-THERE ARE SEVERAL REQUESTS BEFORE YOURS"
   .
   .
   .
REPLY/TO=5
 31-DEC-1988 10:20:25.50, request 5 completed by operator OPA0
```

In this example the OPCOM message indicates that a user wants the operator to place the disk volume labeled OPGUIDE on the disk drive DBA3 and ready the device. The REPLY/PENDING command indicates that the operator can perform the task but not immediately; the /PENDING qualifier prevents the user from entering other commands until the operator fulfills or aborts the request. After mounting the disk on the drive the operator sends a message indicating that the request has been fulfilled. When no message is specified, OPCOM sends a standard message indicating that the task has been performed.

4

```
%%%%%%%%%%  OPCOM, 31-DEC-1988 10:20:50.39  %%%%%%%%%%
request 5 from user ROBINSON
Please mount volume GRAPHIC_FILES in device _DUA11:
Shelf 4 - slot B
$ REPLY/TO=5 "SUBSTITUTE  DUA4"
```

The REPLY/TO command with the SUBSTITUTE syntax in this example is used in response to a MOUNT/ASSIST request entered by user ROBINSON. The MOUNT device is switched to DUA4, and the user is notified.

5

```
$ REPLY/STATUS
%OPCOM, 31-DEC-1988 10:20:50.39, operator status for operator OPA0
DISKS
```

The REPLY/STATUS command in this example requests that the operator terminal status for terminal OPA0 be displayed. The response from OPCOM indicates that terminal OPA0 is enabled to receive messages from disk devices.

6

```
$ REPLY/BELL/TERMINAL=TTC1: "YOUR FILE HAS COMPLETED PRINTING. BOB S."
```

The REPLY command in this example sends a message to the user logged in at terminal TTC1. When the message is displayed, a bell rings at that terminal.

7
```
$ REPLY/ENABLE
$
$ REPLY/ENABLE
%OPCOM, 31-DEC-1988 10:22:19.75, operator status for operator OPA0
CENTRAL, PRINTER, TAPES, DISKS, DEVICES, CARDS, NETWORK, CLUSTER,
OPER11, OPER12
   .
   .
   .
$ REPLY/DISABLE=(PRINTER, TAPES)
%OPCOM, 31-DEC-1988 10:22:26.07, operator disabled, operator OPA0
```

The REPLY/ENABLE command in this example designates terminal OPA0 to receive messages from all facilities. Later, the REPLY/DISABLE command selectively disables OPA0 from receiving messages pertaining to print devices and tapes.

REQUEST

Displays a message at a system operator's terminal and optionally requests a reply. All messages are logged at the operator's console and in the operator's log file, if that file is initialized.

To use this command, you must start the OPCOM process at boot-time by specifying the DCL command @SYS$SYSTEM:STARTUP OPCOM in the site-specific startup command file, SYS$MANAGER:SYSTARTUP.COM.

FORMAT

REQUEST *"message-text"*

PARAMETER

"message-text"

Specifies the text of the message to be displayed. The string can be up to 128 characters. If the string contains spaces, special characters, or lowercase characters, enclose it in quotation marks ("").

DESCRIPTION

When you use the REQUEST command to send a message to an operator, the message is displayed at the operator terminals specified with the /TO qualifier.

If you specify /REPLY, the message is assigned an identification number, so the operator can respond to the message. The system displays the following message:

```
%OPCOM-S-OPRNOTIF, operator notified, waiting...hh:mm:ss
```

When the operator responds to your request, the system displays a message such as the following:

```
%OPCOM-S-OPREPLY, message text entered by operator
```

If you request a reply, you cannot enter any commands until the operator responds. If you press CTRL/C, the system displays the following message:

```
REQUEST - Enter message or cancel with ^Z
REQUEST - Message?
```

At this time, you can either enter another message, or press CTRL/Z to cancel the request. If you enter another message, that message is sent to the operator, and you must continue to wait for a reply.

All messages are logged at the central operator's console and in the system operator's log file, if that file is initialized.

QUALIFIERS

/REPLY

Requests a reply to the message and issues a unique identification number to which the operator sends the response. The system displays a message that the operator has been notified; you cannot enter any commands until the operator responds. If you press CTRL/C before the operator responds, you can then enter another message to the operator, or press CTRL/Z to cancel the request.

/TO=(operator[,...])

Specifies one or more operators to whom you want to send the message. Possible keywords are as follows:

CARDS	Sends the message to operators designated to respond to card reader requests
CENTRAL	Sends the message to the central system operator
CLUSTER	Sends the message to operators designated to respond to cluster-related requests
DEVICES	Sends the message to operators who mount and dismount disks
DISKS	Sends the message to operators who mount and dismount disk volumes
NETWORK	Sends the message to the network operator
OPER1 through OPER12	Sends the message to operators identified as OPER1 through OPER12
PRINTER	Sends the message to operators designated to handle print requests
SECURITY	Sends the message to operators designated to respond to security-related requests
TAPES	Sends the message to operators designated to mount and dismount tape volumes

EXAMPLES

1

```
$ PRINT/COPIES=2/QUEUE=LQ_PRINT  REPORT.OUT/FORM=LETTER
  Job REPRT (queue LQA1, entry 401) pending
$ REQUEST/REPLY/TO=PRINTER -
$_"Have queued job 401 as FORM=LETTER;  can you print it?"
%OPCOM-S-OPRNOTIF, operator notified, waiting...10:42:16.10
%OPCOM-S-OPREPLY, AFTER 11:00
 15-APR-1988 10:25:32.40, request 3 completed by operator OPA0
```

In this example the PRINT command requests that multiple copies of a file be printed using a special paper (/FORM=LETTER). After queueing the job to the printer, the REQUEST command sends a message to the system operator.

The operator sends a reply after completing the request.

2

```
$ REQUEST/REPLY  "Are you there?"
%OPCOM-S-OPRNOTIF, operator notified, waiting...14:54:30.33
CTRL/C
REQUEST-Enter message or cancel request with ^Z
REQUEST-Message?CTRL/Z
%OPCOM-S-OPRNOTIF, operator notified, waiting... 14:59:01.38
%OPCOM-F-RQSTCAN, request was cancelled
```

In this example the REQUEST command issues a message and requests a response. When no operator replies to the question, CTRL/C is used to interrupt the request; then CTRL/Z is used to cancel it.

RETURN

Terminates a GOSUB subroutine procedure and returns control to the command following the calling GOSUB command.

FORMAT

RETURN *[status-code]*

PARAMETER

status-code

Defines a longword (integer) value or expression equivalent to an integer value that gives the exit status of the subroutine by defining a numeric value for the reserved global symbol $STATUS. The value can be tested by the next outer command level. The low-order three bits of the longword integer value change the value of the reserved global symbol $SEVERITY. If you specify a status code, DCL interprets the code as a condition code. Note that even numeric values produce warning, error, and fatal error messages, and that odd numeric values produce either no message or a success or informational message.

If you do not specify a status-code, the current value of $STATUS is saved. When control returns to the outer command level, $STATUS contains the status of the most recently executed command or program.

DESCRIPTION

The RETURN command terminates the GOSUB subroutine and returns control back to the command following the calling GOSUB command.

When a DCL command, user program, or command procedure completes execution, the command interpreter saves the condition code value in the global symbol $STATUS. The system maintains this value in hexadecimal. If a RETURN command does not explicitly set a value for $STATUS, the command interpreter uses the current value of $STATUS to determine the error status.

The low-order three bits of the status value contained in $STATUS represent the severity of the condition. The reserved global symbol $SEVERITY contains this portion of the condition code. Severity values range from zero through four, as shown in the following table:

Value	Severity
0	Warning
1	Success
2	Error
3	Information
4	Severe (fatal) error

Note that the success and information codes have odd numeric values, and that warning and error codes have even numeric values.

EXAMPLE

```
$ SHOW TIME
  18-APR-1988 14:25:42
$ GOSUB SYMBOL
$ EXIT
$ SYMBOL:
$     SHOW SYMBOL RED
      RED = "SET DEFAULT [JONES.DCL]"
$     RETURN 1
```

The GOSUB command transfers control to the subroutine labeled SYMBOL. After the subroutine is executed, the RETURN command transfers control back to the command following the calling GOSUB statement, giving $STATUS and $SEVERITY a value of 1. The procedure then exits.

RUN (Image)

Executes an image within the context of your process. You can abbreviate the RUN command to a single letter, **R**.

If you specify an image name in the command line with an explicit version number (or a semicolon), the image runs with current process privileges. If you do not specify an explicit version number (or semicolon), the image runs with any privileges with which it was installed. If you have DECnet software installed and want to execute an image over the network, you must have READ access to the file.

FORMAT

RUN *file-spec*

PARAMETER

file-spec

Specifies an executable image to be executed. The file type defaults to EXE. Wildcard characters are not allowed.

QUALIFIER

/DEBUG
/NODEBUG

Executes the image under control of the debugger. The default is /DEBUG if the image is linked with /DEBUG and /NODEBUG if the image is linked without /DEBUG. The /DEBUG qualifier is invalid if the image is linked with /NOTRACEBACK. The /NODEBUG qualifier overrides the effect of LINK /DEBUG. If the image was linked with /TRACEBACK, traceback reporting is performed when an error occurs.

If the image was not linked with the debugger, you can specify /DEBUG to request the debugger at execution time. However, if /NOTRACEBACK was specified when the image was linked, the /DEBUG qualifier is invalid.

EXAMPLES

1

```
$ RUN LIBRA
```

The image LIBRA.EXE starts executing in the process. If the image LIBRA has been installed with amplified privileges, it runs with those privileges because you have not explicitly specified a version number or a semicolon. Alternatively, the image LIBRA.EXE still runs with its amplified privileges, if you enter the RUN command as follows:

```
RUN LIBRA.EXE
```

2

```
$ MACRO/ENABLE=DEBUG ORION
$ LINK/DEBUG ORION
$ RUN ORION

     VAX DEBUG Version 4.4

%DEBUG-I-INITIAL, language is MACRO, module set to 'ORION'
DBG>
    .
    .
    .
$ RUN/NODEBUG ORION
```

A program is compiled, linked, and run with the debugger. Subsequently, a RUN/NODEBUG command requests that the debugger, which is present in the image, not issue a prompt. If an error occurs while the image executes, the debugger can perform traceback and report on the error.

3

```
$ RUN AQUARIUS.EXE;1
```

The image AQUARIUS.EXE starts executing in the process. If the image AQUARIUS has been installed with amplified privileges, it does not run with those privileges because you have specified a version number. Instead, the image runs with only current process privileges. When you specify a version number (or even just a semicolon), the image activator does not search its list of special images that have been installed with privileges. The process AQUARIUS still runs with only normal process privileges, if you enter the RUN command as follows:

```
RUN AQUARIUS.EXE;
```

In this case, however, the highest version of the image AQUARIUS runs.

RUN (Process)

Creates a subprocess or a detached process to run an image and deletes the process when the image completes execution. A subprocess is created if any of the qualifiers except /UIC or /DETACHED is specified. A detached process is created if the /UIC qualifier is specified and you have the DETACH user privilege.

FORMAT

RUN *file-spec*

PARAMETER

file-spec

Specifies the file name of an executable image to be executed in a separate process. The default file type is EXE. Wildcard characters are not allowed in the file specification.

DESCRIPTION

The RUN command creates a process to execute the specified image. If you specify the /UIC qualifier, RUN creates a detached process. Otherwise, the RUN command creates a subprocess.

When you specify any qualifiers with the RUN command, the RUN command creates a process and displays the process identification code (PID) in SYS$OUTPUT. The newly created process executes the image named in the file specification. When the image has finished executing, the system deletes the process that was running that image.

By default, the RUN command creates a subprocess with the same UIC (user identification code), current disk and directory defaults, privileges, and priority as the current process.

Both the /DETACHED and the /UIC qualifiers request the RUN command to create a detached process. You must have the user privilege DETACH to create a detached process with a different UIC. When you create a detached process, the resource quotas are the same as those of the current process. However, if you have DETACH privilege, you can specify any quotas for the detached process.

Input, Output, and Error Streams

Use the following qualifiers to assign equivalence names for the logical names SYS$INPUT, SYS$OUTPUT, and SYS$ERROR for the process:

/INPUT
/OUTPUT
/ERROR

The equivalence names you specify for these process-permanent files are interpreted within the context of the process you are creating. For example, file type defaults, and logical name use and translation are image- and language-dependent.

Defining Process Attributes

Use the following qualifiers to override the default attributes for a process:

/ACCOUNTING
/DUMP
/PRIORITY
/PRIVILEGES
/PROCESS_NAME
/SERVICE_FAILURE
/SWAPPING

Assigning Resource Quotas

When you enter a RUN command to create a process, you can define quotas to restrict the amount of various system resources available to the process. The following resource quota is deductible when you create a subprocess; that is, the value you specify is subtracted from your current quota and given to the subprocess:

Qualifier	Quota
/TIME_LIMIT	CPUTIME

The quota amount is returned to your current process when the subprocess is deleted.

The system defines minimum values for each specifiable quota. If you specify a quota that is below the minimum, or if you specify a deductible quota that reduces your current quota below the minimum, the RUN command cannot create the process. To determine your current quotas, enter the SHOW PROCESS/QUOTAS command.

You can also specify limits for nondeductible quotas. Nondeductible quotas are established and maintained separately for each process and subprocess. The following qualifiers specify nondeductible quotas:

Qualifier	Quota
/AST_LIMIT	ASTLM
/EXTENT	WSEXTENT
/IO_BUFFERED	BIOLM
/IO_DIRECT	DIOLM
/MAXIMUM_WORKING_SET	WSQUOTA
/WORKING_SET	WSDEFAULT

A third type of quota treatment is pooling. Pooled quotas are established when a detached process is created. They are shared by that process and all its descendent subprocesses. Charges against pooled quota values are subtracted from the current available totals as they are used and are added back to the total when they are not being used. The following qualifiers specify pooled quotas:

Qualifier	Quota
/BUFFER_LIMIT	BYTLM
/ENQUEUE_LIMIT	ENQLM
/FILE_LIMIT	FILLM
/PAGE_FILE	PGFLQUOTA
/QUEUE_LIMIT	TQELM
/SUBPROCESS_LIMIT	PRCLM

Hibernation and Scheduled Wakeups

Use the following qualifiers to schedule execution of the image:

/DELAY
/INTERVAL
/SCHEDULE

If you specify any of these qualifiers, the RUN command creates the process and places it in hibernation. The process cannot execute the image until it is awakened. Time values specified with these three qualifiers control when the process is awakened to execute the specified image.

You can schedule wakeups for a specified delta time (/DELAY qualifier) or an absolute time (/SCHEDULE qualifier). You can also schedule wakeups for recurrent intervals with the /INTERVAL qualifier. If you specify an interval time, the created process is awakened to execute the specified image at fixed time intervals. If the image terminates normally (for example, by a RET instruction), the process returns to a state of hibernation, awaiting the next scheduled wakeup time and user-mode exit handlers are not called. At the next wakeup time, the image is recalled at its entry point; the image is not reactivated. If the image terminates abnormally, or by an $EXIT command, or by a $FORCEX command, the process does not return to hibernation, further scheduling requests are terminated, user-mode exit handlers are called, the image exits, and the created process is deleted.

Use the /PROCESS_NAME qualifier to give the created process a name. You can use this process name in a subsequent STOP or CANCEL command. A STOP command terminates execution of the image in the process and causes the process to be deleted. The CANCEL command cancels wakeup requests that are scheduled but have not yet been delivered.

QUALIFIERS

/ACCOUNTING (default)
/NOACCOUNTING

Requires ACNT privilege to disable accounting.

Logs accounting records in the system accounting file for the created process.

/AST_LIMIT=quota

Specifies the maximum number of asynchronous system traps (ASTs) that the created process can have outstanding.

If you do specify an AST limit quota, the default quota established at system generation time is used. The minimum required for any process to execute is 2. A value of 10 is typical.

The AST limit quota is nondeductible.

/AUTHORIZE
/NOAUTHORIZE (default)

Requires DETACH privilege.

When the image to be executed is the system login image (LOGINOUT.EXE), this qualifier searches the user authorization file to validate a detached process. The /NOAUTHORIZE qualifier creates a detached process that runs under the control of the command interpreter.

Specify /AUTHORIZE if you want the login image to check the user authorization file whenever a detached process is created. The process-permanent files specified by the /INPUT and /OUTPUT qualifiers are made available to the command interpreter for input and output.

Any nonspecified attributes of the created process default to those of the current process.

/BUFFER_LIMIT=quota

Specifies the maximum amount of memory, in bytes, that the process can use for buffered I/O operations or for temporary mailbox creation.

If you do not specify a buffered I/O quota, the default value established at system generation time is used. The minimum amount required for any process to execute is 1024 bytes. A value of 10,240 is typical.

The buffer limit quota is pooled.

/DELAY=delta-time

Places the created process in hibernation and awakens it after a specified time interval.

Specify the delta time according to the rules for entering delta times given in Section 1.4 of the *VMS DCL Concepts Manual*.

If you specify both /DELAY and /INTERVAL, the first wakeup request occurs at the time specified by /DELAY. All subsequent wakeups occur at the interval specified by /INTERVAL.

/DETACHED
/NODETACHED

Creates a detached process with the same user identification code (UIC) as the current process. (To create a detached process with a different UIC, use the /UIC qualifier.) By default, the created process is not a detached process.

By default, the detached process has the same resource quotas as the current process; the DETACH privilege allows you to specify any quotas you need for the detached process. Unless you have the DETACH privilege, the maximum number of detached processes that you can create is limited to the quota defined by MAX_DETACH in your user authorization file.

/DUMP
/NODUMP (default)

When an image terminates because of an unhandled error, /DUMP causes the contents of the address space to be written to the file named SYS$LOGIN:IMAGEDUMP.DMP. You can then use the Analyze/Process_Dump Utility to analyze the dump.

/ENQUEUE_LIMIT=quota

Specifies the maximum number of locks that a process can have outstanding at any one time.

The default quota is that established at system generation time. The minimum required for any process to operate is 2. A value of 6 is typical.

/ERROR=file-spec

Defines an equivalence name string of 1 to 63 alphanumeric characters for the logical device name SYS$ERROR. The logical name and equivalence name are placed in the process logical name table for the created process. (The /ERROR qualifier is ignored if you are running SYS$SYSTEM:LOGINOUT.)

/EXTENT=quota

Specifies the maximum size to which the image being executed in the process can increase its physical memory size.

The default quota is that established at system generation time. The minimum value required for any process to execute is 10 pages. A value in the range of 400 to 2000 is typical. The extent quota is nondeductible.

/FILE_LIMIT=quota

Specifies the maximum number of files that a process can have open at any one time.

The default quota is the quota established at system generation time. The minimum amount required for any process to execute is 2. A value of 20 is typical. The file limit quota is pooled.

/INPUT=file-spec

Defines an equivalence name string of 1 to 63 characters for SYS$INPUT. The logical name and equivalence name are placed in the process logical name table for the created process.

/INTERVAL=delta-time

Requests that the created process be placed in hibernation and awakened at regularly scheduled intervals.

Specify the delta time according to the rules for entering delta times given in Section 1.4 of the *VMS DCL Concepts Manual*.

If you specify the /DELAY or /SCHEDULE qualifier with the /INTERVAL qualifier, the first wakeup occurs at the time specified by /DELAY or /SCHEDULE; all subsequent wakeups occur at intervals specified by /INTERVAL. If you specify neither /DELAY nor /SCHEDULE with /INTERVAL, the first wakeup occurs immediately by default.

/IO_BUFFERED=quota

Specifies the maximum number of system-buffered I/O operations that the created process can have outstanding at any one time.

The default quota is the quota established at system generation time. The minimum required for any process to execute is 2. A value of 6 is typical. The buffered I/O quota is nondeductible.

/IO_DIRECT=quota

Specifies the maximum number of direct I/O operations that the created process can have outstanding at any one time.

The default quota is the quota established at system generation time. The minimum required for any process to execute is 2. A value of 6 is typical. The direct I/O quota is nondeductible.

/JOB_TABLE_QUOTA=quota

Allows you to specify a quota for a detached process's jobwide logical name table.

A value of 0 has a special meaning. It means that the table, for all practical purposes, has infinite quota because its quota is pooled with that of its parent table, the system directory table.

Note that the /JOB_TABLE_QUOTA qualifier is relevant only for detached processes. If the /JOB_TABLE_QUOTA qualifier is specified in a RUN command which results in the creation of a subprocess, it is ignored.

/MAILBOX=unit

Specifies the unit number of a mailbox to receive a termination message when the created process is deleted. If no mailbox is specified, the creating process receives no notification when the subprocess or detached process has been deleted.

/MAXIMUM_WORKING_SET=quota

Specifies the maximum size to which the image being executed in the process can increase its working set size. An image can increase its working set size by calling the Adjust Working Set Limit system service.

The default quota is the quota established at system generation time. The minimum value required for any process to execute is 10 pages. A value of 200 is typical.

The maximum working set quota is nondeductible.

/OUTPUT=file-spec

Defines an equivalence name string of 1 to 63 characters for the logical device name SYS$OUTPUT. Both the equivalence name and the logical name are placed in the process logical name table for the created process.

/PAGE_FILE=quota

Specifies the maximum number of pages that can be allocated in the paging file for the process. The paging file quota is the amount of secondary storage available during execution of the image.

The default quota is the quota established at system generation time. The minimum value required for a process to execute is 256 pages. A value of 10,000 pages is typical. The paging file quota is pooled.

/PRIORITY=n

Requires ALTPRI privilege to set the priority higher than your current process.

Specifies the base priority at which the created process executes.

The value of *n* is a decimal number from 0 through 31, where 31 is the highest priority and 0 is the lowest. Normal priorities range from 0 through 15; real-time priorities range from 16 through 31.

The default priority is that of the current process.

/PRIVILEGES=(privilege[,...])

Requires SETPRV privilege to specify privileges that you do not have.

Defines user privileges for the created process. You can extend any privilege you possess to a process you create. By default, the created process has the same privileges as its creator. If you specify only one privilege, you can omit the parentheses.

For a list of process privileges, see Table A-1 in the *VMS DCL Concepts Manual*.

You can also use the keyword NOSAME as the privilege parameter. If you specify /PRIVILEGES=NOSAME, the created process has no privileges.

If you specify a version number (or semicolon) in the file-spec parameter, the current process privileges are used, overriding any privileges specified with the /PRIVILEGES qualifier.

/PROCESS_NAME=process-name

Specifies a name of 1 to 15 characters for the created process. The process name is implicitly qualified by the group number of the process's user identification code (UIC). By default, the name is null.

/QUEUE_LIMIT=quota

Specifies the maximum number of timer queue entries that the created process can have outstanding at any one time. This number includes timer requests and scheduled wakeup requests.

The default quota is the quota established at system generation time. A process does not require any timer queue quota in order to execute. A value of 8 is typical.

The timer queue entry quota is pooled.

/RESOURCE_WAIT (default)
/NORESOURCE_WAIT

Places the created process in a wait state when a resource required for a particular function is not available.

If you specify /NORESOURCE_WAIT, the process receives an error status code when a resource is unavailable.

/SCHEDULE=absolute-time

Places the created process in hibernation and awakens it at the specified time.

Specify the absolute time value according to the rules for entering absolute time values given in Section 1.4 of the *VMS DCL Concepts Manual*.

/SERVICE_FAILURE
/NOSERVICE_FAILURE (default)

Enables or disables an exception condition notification if an error occurs during a system service request. By default, an error status code is returned to the process.

If you specify /SERVICE_FAILURE and an error occurs during a system service request, the process encounters an exception condition.

/SUBPROCESS_LIMIT=quota

Specifies the maximum number of subprocesses that the created process is allowed to create.

The default quota is the quota established at system generation time. A process does not require any subprocess quota in order to execute. A value of 8 is typical.

The subprocess limit quota is pooled.

/SWAPPING (default)
/NOSWAPPING

Requires PSWAPM privilege to inhibit process swapping.

Permits the process to be swapped. The default allows a process to be swapped from the balance set in physical memory to allow other processes to execute.

With /NOSWAPPING in effect, the process is not swapped out of the balance set when it is in a wait state. By default, a process may be swapped out of the balance set whenever it is in a wait state.

/TIME_LIMIT=limit

Specifies the maximum amount of CPU time (in delta time) a created process can use. CPU time is allocated to the created process in units of 10 milliseconds. When it has exhausted its CPU time limit quota, the created process is deleted.

If this quota is not specified and the created process is a detached process, the detached process receives a default value of 0, that is, unlimited CPU time.

If this quota is not specified and the created process is a subprocess, the subprocess receives half the CPU time limit quota of the creating process.

If this quota is specified as 0, the created process has unlimited CPU time providing that the creating process also has unlimited CPU time. If, however, the creating process does not have unlimited CPU time, the created process receives half the CPU time limit quota of the creating process.

The CPU time limit quota is a consumable quota; that is, the amount of CPU time used by the created process is not returned to the creating process when the created process is deleted.

If you restrict CPU time for a process, specify the time limit according to the rules for specifying delta time values, as given in Section 1.4 of the *VMS DCL Concepts Manual*.

/UIC=uic

Specifies that the created process be a detached process and assigns it a user identification code (UIC). Specify the UIC using standard UIC format as described in Section 8.1 of the *VMS DCL Concepts Manual*.

/WORKING_SET=default

Specifies the number of pages in the working set of the created process.

The default working set size is the size established at system generation time. The minimum number of pages required for a process to execute is 10 pages. The value specified cannot be greater than the quota specified with /MAXIMUM_WORKING_SET. A value of 200 pages is typical.

The maximum working set quota is nondeductible.

EXAMPLES

1
```
$ RUN/PROCESS_NAME=SUBA   SCANLINE
%RUN-S-PROC_ID, identification of created process is 00010044.
```

In this example, the RUN command creates a subprocess named SUBA to run the image SCANLINE.EXE. The system gives the subprocess an identification number of 00010044.

2
```
$ RUN/DELAY=3:30/OUTPUT=BALANCE.OUT BALANCE
```

In this example, the RUN command creates a subprocess to run the image BALANCE.EXE 3 hours and 30 minutes from now; output is written to the file BALANCE.OUT.

3
```
$ RUN/INTERVAL=1:40/PROCESS_NAME=STAT   STATCHK
%RUN-S-PROC_ID, identification of created process is 00050023
   .
   .
   .
$ CANCEL STAT
```

In this example, the RUN command creates a subprocess named STAT to execute the image STATCHK.EXE. The process is scheduled to execute the image at intervals of 1 hour and 40 minutes. The process hibernates; however, because neither the /DELAY nor /SCHEDULE qualifier is specified, the first wakeup occurs immediately.

The CANCEL command subsequently cancels the wakeup requests posted by the /INTERVAL qualifier. If the process is currently executing the image, it completes the execution and hibernates.

4
```
$ RUN/PROCESS_NAME=LYRA  LYRA -
$_/OUTPUT=_TTB3: -
$_/ERROR=_TTB3:
%RUN-S-PROC_ID, identification of created process is 000A002F
```

In this example, the RUN command creates a subprocess named LYRA to execute the image LYRA.EXE. The /OUTPUT and /ERROR qualifiers assign equivalences to the logical names SYS$OUTPUT and SYS$ERROR for the subprocess. Any messages the subprocess writes to its default output devices are displayed on the terminal TTB3.

5
```
$ RUN/UIC=[100,4]/PRIVILEGES=(SAME,NOPSWAPM) -
$_/NORESOURCE_WAIT   OVERSEER
%RUN-S-PROC_ID, identification of created process is 0001002C
```

In this example, the RUN command creates a detached process to execute under the UIC [100,4]. The image OVERSEER.EXE is executed. The RUN command gives the process all the privileges of the current process, except the ability to alter its swap mode. The /NORESOURCE_WAIT qualifier disables resource wait mode for the process.

6
```
$ DEFINE/GROUP  TEST [MALCOLM.TESTFILES]
$ RUN/PROCESS=SUB  WATCH -
$_/INPUT=TEST:OUT1 -
$_/OUTPUT=F$LOGICAL("SYS$OUTPUT")
%RUN-S-PROC_ID, identification of created process is 0001002E
```

In this example, the DEFINE command creates an entry in the group logical name table for the logical name TEST. The RUN command creates a subprocess to execute the image WATCH.EXE.

The /INPUT qualifier defines SYS$INPUT for the subprocesses. The logical name TEST defines the directory for the file OUT1.DAT. Because the logical name TEST is in the group logical name table, the logical name can be translated and referred to by the image WATCH.EXE.

The /OUTPUT qualifier uses the lexical function F$LOGICAL to translate the logical name of the current process's SYS$OUTPUT device. The equivalence name string is equated to the device SYS$OUTPUT for the subprocess.

RUNOFF

Invokes the DIGITAL Standard Runoff (DSR) text formatter to format one or more ASCII files. Creates formatted files from source DSR (RNO) files, unformatted table of contents (RNT) files, and unformatted index (RNX) files. Optionally creates intermediate (BRN) files for input to RUNOFF /CONTENTS and RUNOFF/INDEX commands.

For a complete description of the DSR formatter, including more information about the RUNOFF command, see the *VAX DIGITAL Standard Runoff Reference Manual.* For information about the DCL commands RUNOFF/CONTENTS and RUNOFF/INDEX, see the DCL command descriptions.

FORMAT

RUNOFF *file-spec[,...]*

PARAMETER

file-spec[,...]

Specifies one or more ASCII files (containing text and DSR commands) to be formatted by the RUNOFF command. The input file type defaults to RNO; you must specify the file type for RNT and RNX files. Separate multiple files with commas. Wildcard characters are not allowed in the file specification.

DSR produces an output file having the same file name as the input file. The output file type depends on the input file type. The default output file type is MEM. For a list of input file types and the associated output file types, see the *VAX DIGITAL Standard Runoff Reference Manual.*

Specify SYS$INPUT to type the input from your terminal or a command procedure; terminate input from the terminal by pressing CTRL/Z.

DESCRIPTION

The RUNOFF command allows you to do the following:

- Adjust the amount of text on a page
- Control the position of text on a page
- Control underlining, overwriting, and bolding of text
- Override some DSR commands and flags in your input file
- Process all or part of your input file
- Create an intermediate file for indexes or tables of contents

QUALIFIERS

/BACKSPACE

Positional qualifier.

Controls whether DSR uses the ASCII backspace character to perform character-by-character overprinting. By default, DSR performs line-by-line overprinting.

/BOLD[=n]
/NOBOLD

Positional qualifier.

Specifies the number of times characters are overstruck in a bolding operation. You can specify the number of times DSR overprints flagged text by stating a value for n. N must be 0 or a positive integer and defaults to 1. A specification of /BOLD=0 or /NOBOLD disables all bolding, even if the appropriate flags are recognized and enabled.

/CHANGE_BARS["character"]
/NOCHANGE_BARS

Positional qualifier.

Controls whether DSR generates change bars in the formatted file. The default change-bar character is the vertical bar (|). The change bars appear 3 spaces to the left of the lines of text that you have marked for change bars.

See the .BEGIN BAR and .END BAR commands in the *VAX DIGITAL Standard Runoff Reference Manual.*

You can replace the default change-bar character by supplying a substitute character for the /CHANGE_BARS["character"] qualifier. You must specify the replacement character as either a character enclosed in quotation marks or as an octal, decimal, or hexadecimal value for the desired character.

The /CHANGE_BARS qualifier without a value uses the default change-bar character (|). The /NOCHANGE_BARS qualifier overrides any change-bar commands in the input file and disables the output of change bars.

/DEBUG[=(option[,...])]
/NODEBUG (default)

Positional qualifier.

Traces certain operations by placing the DSR commands in the output file. The options are as follows:

ALL	Specifies all five options (CONDITIONALS, CONTENTS, FILES, INDEX, and SAVE_RESTORE).
CONDITIONALS	Causes DSR to ignore all conditional processing commands (.IF, .IFNOT, .ELSE, .ENDIF) in the input file. DSR includes both "true" and "false" conditional information in the output file along with formatted text. See the *VAX DIGITAL Standard Runoff Reference Manual* for further details on the .IF, .IFNOT, .ELSE, .ENDIF, and .VARIABLE commands and the /VARIANT qualifier.
CONTENTS	Causes DSR to output all .SEND TOC commands along with the text being sent to the table of contents.
FILES	Causes DSR to output all .REQUIRE commands as well as the text of the require files.
INDEX	Causes DSR to output the indexing commands, .INDEX and .ENTRY, in addition to the text to which they refer.
SAVE_RESTORE	Causes DSR to output all .SAVE and .RESTORE commands.

If you specify more than one option, separate them with commas and enclose the list in parentheses. If you specify /DEBUG without specifying any options, ALL is assumed.

/DEVICE=(option[,...])

Positional qualifier.

Controls whether DSR generates an output file (LNI) that is suitable for printing on an LN01, LN01E, or an LN03 laser printer.

If you do not get the output that you expect when you print an LNI file on an LN01 or an LN01E, check with your system manager. The *VAX DIGITAL Standard Runoff Reference Manual* contains information for system managers about setting LN01 and LN01E laser printers to print LNI files.

You can choose options from the following list to indicate output device, page orientation, and type of emphasis for flagged characters in your LNI file:

LN01	Produces an output file suitable for printing on an LN01 laser printer; the default paper size is 8 1/2 by 11 inches; the default mode is PORTRAIT. The output file name is the same as the input file name; the default file type is LNI.
LN01E	Produces an output file suitable for printing on an LN01E laser printer using the standard European paper size (A4). The output file name is the same as the input file name. The default file type is LNI; the default mode is PORTRAIT. Incompatible with LN01.
LN03	Produces an output file suitable for printing on an LN03 laser printer; the default paper size is 8 1/2 x 11 inches. The output file name is the same as the input file name. The default file type is LNI; the default mode is PORTRAIT.
LANDSCAPE	Causes the appropriate fonts for landscape mode to be loaded into the LN01; pages are printed with the long dimension at top using a smaller type size. (The page is 11 inches wide and 8 1/2 inches long.) Allowable page dimensions are 0 to 73 lines per page and 0 to 132 characters per line. Incompatible with PORTRAIT.
PORTRAIT (default)	Causes the appropriate fonts for portrait mode to be loaded into the LN01; pages are printed with the short dimension at top using a larger type size. (The page is 8 1/2 inches wide and 11 inches long.) Allowable page dimensions are 0 to 66 lines per page and 0 to 80 characters per line. Incompatible with LANDSCAPE. PORTRAIT mode is the default when you specify /DEVICE=LN01, /DEVICE=LN01E, or /DEVICE=LN03.

ITALIC (default)	Causes the italic and bold-italic fonts to be loaded into the LN01 printer. Italicizes characters flagged for underlining. Italicized characters can also be bolded depending on the type of emphasis you specify in your input file. The LN03 requires no loading of fonts since default fonts are present. Text flagged for emphasis is printed italic if the current font has the ITALIC attribute; otherwise the flagged text is underlined.
UNDERLINE	Causes the text and bold fonts to be loaded into the LN01. Underlines characters flagged for underlining. The LN01 allows only 63 consecutive characters (counting a space as a character) to be underlined per line. If you want to underline individual words and not the spaces between them, you will be able to underline only 63 words per line. Incompatible with ITALIC. DSR does not report an error if the user exceeds this limit of the hardware. On an LN03, if you have specified the underlining option the flagged text is underlined. The printer does not default to italic even if the current font has the ITALIC attribute.

/DOWN[=n]
/NODOWN (default)

Positional qualifier.

Controls whether DSR inserts a specified number of blank lines at the top of each page. These blank lines precede any header information. The number of blank lines you specify (n) does not affect the number of text lines on a page.

For example, if you specify /DOWN=10 with a .PAGE SIZE of 58 lines, up to 58 lines of text will be output after 10 blank lines.

If you specify the /DOWN qualifier without a value, five blank lines are inserted. If you specify /DOWN=0 or omit the qualifier, no blank lines are inserted, except those associated with the print device or header layout.

/FORM_SIZE=n

Specifies the maximum number of lines per page including running heads and running feet. Defaults to /FORM_SIZE=66, which is standard for 11-inch paper. For laser printers, set the number of lines as follows:

Paper size	Lines	Mode
8.05	69	Landscape
8.28	71	Landscape (LN01E default)
8.51	73	Landscape (LN01, LN03 default)
11.00	66	Portrait (LN01, LN03 default)
11.66	70	Portrait (LN01E default)

Paper size	Lines	Mode
12.33	74	Portrait
13.00	78	Portrait
14.00	84	Portrait

When used with /SIMULATE, /FORM_SIZE controls the physical size of the page by putting out line feeds to match the number specified by the value n. When used with /NOSIMULATE, /FORM_SIZE=n causes DSR to suppress the form feed that DSR would normally insert at the line number specified by the value n. If the number of lines that DSR is going to put on any given page does not match the value n, a form-feed character is written into the output file.

The default value for the value n is derived from the VAX Run-Time Library (RTL) routine LIB$LP_LINES. This defaults to 66 unless the logical SYS$LP_LINES is defined, in which case, the assigned value is used. To change the default value, specify a different value for /FORM_SIZE=n.

/INTERMEDIATE[=file-spec]
/NOINTERMEDIATE (default)

Positional qualifier.

Controls whether DSR generates an intermediate output file that can be used as input to the DSR table of contents utility and the DSR indexing utility. See the descriptions of RUNOFF/CONTENTS and RUNOFF/INDEX for more information on producing tables of contents and indexes.

If you specify /INTERMEDIATE, DSR creates an output file that has the same file name as the input file and a file type of BRN. To rename the output file, supply a file specification that is different from the default values.

/LOG
/NOLOG (default)

Controls whether a termination message is displayed at the terminal after successful completion of the DSR operation. The message states the DSR version number, the number of diagnostic messages (if any), the number of output pages, and the output file specification.

If /INTERMEDIATE is specified, the message also includes the number of index records produced and the number of table of contents records produced.

If there are errors in processing, DSR displays a message on the terminal even if the /NOLOG qualifier is specified.

/MESSAGES=(option[,...])

Positional qualifier.

Specifies the destination of all DSR error messages. To indicate a specific destination, use one or both of the following options:

OUTPUT	Messages are sent to the output MEM file
USER	Messages are displayed on the terminal (SYS$ERROR)

If you specify both options, separate them with commas and enclose the list in parentheses.

The default, /MESSAGES=(OUTPUT,USER), sends messages to the output MEM file and displays them on the terminal.

/OUTPUT[=file-spec]
/NOOUTPUT

Positional qualifier.

Specifies that an output file is to be produced and optionally names it. If you specify /OUTPUT without a file specification, or if you omit the qualifier, the directory and file name default to that of the DSR file. If you specify /NOOUTPUT, no output file is produced. The output file type depends on the input file type. The default input file type is RNO and the default output file type is MEM.

The file type defaults to one of the following:

BLB	For an RNB input file
CCO	For an RNC input file
DOC	For an RND input file
ERR	For an RNE input file
HLP	For an RNH input file
LNI	For an RNO input file with /DEVICE set to LN01, LN01E, or LN03
MAN	For an RNM input file
MEC	For an RNT input file
MEM	For an RNO input file with no /DEVICE specification
MEX	For an RNX input file
OPR	For an RNP input file
PLM	For an RNL input file
STD	For an RNS input file

For a complete list of input file types and the associated output file types, see the *VAX DIGITAL Standard Runoff Reference Manual.*

To change the name of the output file, supply a file specification for the value file-spec.

/OUTPUT=SYS$OUTPUT causes output to be sent to the terminal rather than to a disk file. You can use the value SYS$OUTPUT when you are logged in to a hard-copy terminal device of the "daisy wheel" type.

The /NOOUTPUT qualifier suppresses the creation of an output file. Using the /NOOUTPUT qualifier with the /INTERMEDIATE qualifier causes DSR to produce only an intermediate BRN file and not a formatted output file.

You can also use /NOOUTPUT to check an input file for errors without using system resources to generate a formatted output file.

/PAGES=string

Positional qualifier.

Specifies that only the pages within the specified range be generated as output. By default, DSR generates output for all pages. Specify the range as follows:

```
start-page-no:end-page-no,...
```

You can specify up to five ranges (/PAGES="2-9:2-12, 4-1:4-10, 5-9:5-9, A-1:A-3, Index-1:Index-5"). You can omit the colon and the end page number on the last range. You can omit the quotation marks if you specify only one range. Page numbers must be specified in their default form, not the form specified in a .DISPLAY command. You can specify just the appendix letter or name to produce an entire appendix. You can specify just the word INDEX to produce an entire index.

If you specify only a starting page number, output begins at the specified page and continues to the end of the file. To output a single page, the start range and end range must be the same (/PAGES=5:5).

For an entire appendix, only a letter is required (for example, /PAGES="A"). For an entire index, only the word "INDEX" is required (/PAGES="INDEX"). You can refer to specific appendix or index pages with a numeric suffix such as INDEX-10.

Note that the /PAGES qualifier does not recognize any display modes. You must specify the default form of page numbers (5-15) rather than any special form you may have specified with the .DISPLAY NUMBER command (V-15). For details on .DISPLAY NUMBER, see the *VAX DIGITAL Standard Runoff Reference Manual.*

/PAUSE
/NOPAUSE (default)

Controls whether DSR pauses after printing each page of output.

You can use the /PAUSE qualifier to insert single sheets of paper or reproduction masters into hardcopy output devices. When output is halted, the terminal bell rings to remind you to insert a new form. Press the space bar to resume processing.

Do not use this qualifier in a batch job.

/REVERSE_EMPHASIS

Positional qualifier.

Directs DSR to change the order in which flagged text is underlined on an output device. If you use this qualifier, the printer first prints the characters to be underlined, then issues a carriage return without a line feed, and prints the underscores to underline the flagged text. If you view your file on the terminal, the flagged text is overwritten by the underline character.

/RIGHT[=n]
/NORIGHT (default except for LN01)

Positional qualifier.

Causes the text on each page (including header information) to be shifted to the right the number of columns specified by n. This qualifier does not affect the page width. If you specify /RIGHT without specifying a number, text is shifted to the right five spaces. If you specify a value of zero or omit the qualifier, no shift occurs.

The defaults (if /RIGHT is not specified) for LN01 files are as follows:

Mode	LN01	LN01E	LN03
Landscape	9	13	9
Portrait	2	2	2

/SEPARATE_UNDERLINE[="character"]

Positional qualifier.

Prints underlines as separate characters on the next line instead of overstriking with underscores on the same line. The value specifies the character to be used for the underline character and defaults to a hyphen (-). You can specify the underline character as a single printable character or a number preceded by a radix indicator (%D, %O, or %X) to represent the ASCII value of a printable or nonprintable character.

Incompatible with the /[NO]UNDERLINE_CHARACTER qualifiers.

/SEQUENCE
/NOSEQUENCE (default)

Positional qualifier.

Controls whether DSR precedes the lines in the output file with the line numbers of the corresponding lines in the DSR file.

For editors that generate line numbers in the input file, the /SEQUENCE qualifier causes similar numbering to appear in the output file. The line numbers appear in the left margin at the beginning of each line of output.

If the text editor does not generate sequential numbers in the input file, sequential numbers are still generated in the output file, but without leading zeros.

/SIMULATE
/NOSIMULATE (default)

Controls whether DSR uses line feeds or form feeds to advance to the top of each page.

For devices that do not have a form-feed capability, use /SIMULATE to generate enough blank lines to cause a skip to the top of each new page. The /SIMULATE qualifier also causes a pause before the first page of output. To continue after the pause, press the space bar.

/UNDERLINE_CHARACTER[="character"]
/NOUNDERLINE_CHARACTER

Positional qualifier.

Specifies the character to be used for the underline character. Defaults to an underscore (_). You can specify the underline character as a single printable character (enclosed in quotation marks) or as a number preceded by a radix indicator (%D, %O, or %X) to represent the ASCII value of a printable or nonprintable character. A specification of /NOUNDERLINE_CHARACTER overrides any .ENABLE UNDERLINING command in the DSR file. Incompatible with /SEPARATE_UNDERLINE.

/VARIANT=string

Positional qualifier.

Controls the processing of the conditional commands (.IF, .IFNOT, .ELSE, and .ENDIF) by specifying the names of the segments to be processed. See the *VAX DIGITAL Standard Runoff Reference Manual* for descriptions of the conditional commands.

You must name conditional structures introduced by .IF to process them. You must name conditional structures introduced by .IFNOT to exclude them. You must not name conditional structures introduced by .ELSE to process them. If you specify multiple names in a string, separate them by commas and enclose the string in quotation marks.

EXAMPLES

1
```
$ RUNOFF CHAPT1.RNO
```

The RUNOFF command in this example takes the input file, CHAPT1.RNO, and writes formatted output to the file CHAPT1.MEM.

2
```
$ RUNOFF CHAPT1/RIGHT=10,CHAPT2
```

The RUNOFF command in this example produces a CHAPT1.MEM file with margins ten spaces to the right of the margins specified in the input file CHAPT1.RNO. It also generates a CHAPT2.MEM file whose margins are not affected by the /RIGHT=10 qualifier.

3
```
$ RUNOFF/OUTPUT=SYS$OUTPUT TEXT.DAT,INTRO
```

The RUNOFF command in this example sends output to the terminal rather than to a disk file. The qualifier applies to both the input files, TEXT.DAT and INTRO.RNO.

4
```
$ RUNOFF/NOOUTPUT/INTERMEDIATE -
$_CHAPT1,CHAPT2,CHAPT3,CHAPT4,CHAPT5/LOG
```

The RUNOFF command in this example generates intermediate BRN files for each of the input files. The BRN files are used as input for the DSR table of contents program, and for the DSR indexing program. The /NOOUTPUT qualifier suppresses the generation of formatted text files for each input file. The /LOG qualifier produces a termination message after RUNOFF processes each file.

RUNOFF/CONTENTS

Invokes the DIGITAL Standard Runoff (DSR) table of contents utility to create an RNT file that can be processed by DSR to make a table of contents. The input file for this command is an intermediate binary file (BRN) that is produced with the RUNOFF command and the /INTERMEDIATE qualifier (see the RUNOFF command).

For a complete description of the DSR table of contents utility, see the *VAX DIGITAL Standard Runoff Reference Manual.*

FORMAT

RUNOFF/CONTENTS *file-spec[,...] or file-spec[+...]*

PARAMETER

file-spec[,...] or file-spec[+...]

Specifies one or more intermediate binary files (BRN) that contain information (chapter titles, header levels, sections, and so on) for making a table of contents. To create a BRN file, use the RUNOFF command with the /INTERMEDIATE qualifier. See the RUNOFF command for more information on the /INTERMEDIATE qualifier.

If you omit the input file type, the DSR table of contents utility uses a default file type of BRN. RUNOFF/CONTENTS will also process BTC files that the previous version of DSR produced.

For single input files, the table of contents utility produces an output file with the same file name as the input file. The output file type is RNT.

If you separate multiple input files with commas, separate RNT files for each input file are created. If you separate multiple input files with plus signs (+), a single RNT file that contains table of contents information for all of the input files is created. The default output file name is the same as the first input file name; the default file type is RNT. Wildcard characters are not allowed in the file specification.

DESCRIPTION

The RUNOFF/CONTENTS command allows you to create an RNT file that can be processed by DSR to produce a table of contents. Qualifiers to this command allow you to specify the following characteristics for table of contents entries:

- Bolding or underlining of entries
- Deepest header level to be included
- Running page numbers or chapter-oriented page numbers
- Section numbers displayed or not displayed

QUALIFIERS

/BOLD
/NOBOLD (default)

Controls whether the bolding specified in chapter and header titles in the input file appears in the table of contents.

If you specify /BOLD, the text flagged for bolding in the body of the document is marked for overprinting in the finished table of contents.

If you specify /NOBOLD, the text flagged for bolding in the document is not overprinted in the table of contents.

/DEEPEST_HEADER=n

Controls how many levels of header levels are output in the table of contents. You can specify any number of header levels (up to 6) to be displayed by changing the value of n.

The default is /DEEPEST_HEADER=6.

/IDENTIFICATION
/NOIDENTIFICATION (default)

Controls whether the current version number of the DSR table of contents utility is reported.

/INDENT
/NOINDENT (default)

Controls how many spaces the header levels after level 1 are indented in the table of contents.

If you omit this qualifier, or if you specify /NOINDENT, all header levels after header level 1 are indented 2 spaces.

If you specify /INDENT, each header level after header level 1 is indented 2 spaces beyond the preceding header level.

/LOG
/NOLOG (default)

Controls whether the DSR table of contents utility displays the name of each input file as it is processed and after it is processed. The name of each output file created may also be displayed. If there are any errors in processing, the DSR table of contents utility sends messages to the terminal even if /NOLOG is in effect.

/OUTPUT[=file-spec]
/NOOUTPUT

Specifies that an output file is to be produced and optionally names it. If you specify the /OUTPUT qualifier without a file specification, or if you omit the qualifier entirely, the output file name matches the input file name. The default file type is RNT.

You can change the name of the output file by supplying a file specification for the value file-spec.

The /NOOUTPUT qualifier suppresses the creation of an output file. You can use /NOOUTPUT to check an input file for errors without using system resources to generate an output file.

/PAGE_NUMBERS=(option[,...])

Controls whether the page number references in the table of contents are running page numbers or chapter-oriented page numbers; also controls how many levels of headers have page references listed in the table of contents. To specify these options, select from the following list:

Option	Purpose
LEVEL=n	Specifies that header levels up to and including header level n have page numbers listed in the table of contents. The default is to display page numbers for 6 levels of headers.
NORUNNING	Specifies chapter-oriented page numbers (such as 1-3, 10-42). You can specify chapter-oriented numbers for the table of contents even if the document does not have chapter-oriented numbers. NORUNNING is the default.
RUNNING	Specifies running page numbers (such as 3, 42). You can specify running page numbers for the table of contents even if the document does not have running page numbers.

If you supply more than one option, separate them with commas and enclose the list in parentheses.

/REQUIRE=file-spec
/NOREQUIRE (default)

Allows you to change or delete the heading on the first page of a table of contents. The default heading is the word CONTENTS centered on the page and followed by one blank line. You can either substitute another word as a heading, or have no heading.

To change the heading, do one of the following:

- If you do not want any heading, specify a null file as the file specification for /REQUIRE.

  ```
  $ RUNOFF/CONTENTS/REQUIRE=nl:
  ```

- If you want to use a different heading, create or edit a file that specifies the heading that you want. Use the file that you create as the file specification for the /REQUIRE qualifier.

When you use the /REQUIRE qualifier, the default heading for the first page of the contents is not generated. The file that you are "requiring" must provide the heading. The file can contain both DSR commands that change the format of the first page and the text that you want to appear at the top of the page. Or the file can contain only DSR commands to format the first page of the contents. For example, you can put the command .FIGURE 10 in the file. This command generates 10 blank lines at the top of the first page of the table of contents. You can use these blank lines for later pasteup.

/SECTION_NUMBERS (default)
/NOSECTION_NUMBERS

Controls whether the DSR table of contents utility displays section numbers in the table of contents. The /SECTION_NUMBERS qualifier displays section numbers for all header levels in the table of contents. /NOSECTION_NUMBERS suppresses the display of section numbers for all header levels.

/UNDERLINE
/NOUNDERLINE (default)

Controls whether the underlining specified in chapter and header titles in the input file appears in the table of contents.

If you specify /UNDERLINE, the text flagged for underlining in the body of the document is underlined in the table of contents.

If you specify /NOUNDERLINE, the text flagged for underlining in the body of the document is not underlined in the table of contents.

EXAMPLES

1 `$ RUNOFF/INTERMEDIATE CHPT1,CHPT2,CHPT3`

Before using RUNOFF/CONTENTS, you must use RUNOFF/INTERMEDIATE to create a BRN file as input for the DSR table of contents utility. The command line in this example creates three separate files: CHPT1.BRN, CHPT2.BRN, and CHPT3.BRN.

2 `$ RUNOFF/CONTENTS CHPT1.BRN`

In this example, the RUNOFF/CONTENTS command takes the file CHPT1.BRN as input and creates CHPT1.RNT, which can be processed by DSR to produce a final table of contents for Chapter 1.

3 `$ RUNOFF/CONTENTS/INDENT/NOSECTION_NUMBERS CHPT2`

The command in this example takes the file CHPT2.BRN as input and creates CHPT2.RNT. When processed with the RUNOFF command, the RNT file will produce a table of contents in which each header level after header level 1 is indented 2 spaces beyond the preceding header level. The table of contents will not have section numbers listed. See the following example for a sample command line for processing RNT files.

4 `$ RUNOFF/LOG CHPT2.RNT`

The command in this example produces CHPT2.MEC, which is a formatted table of contents. You can use the TYPE or PRINT commands to view the table of contents.

RUNOFF/INDEX

Invokes the DIGITAL Standard Runoff (DSR) indexing utility to create an RNX file that can be processed by DSR to create an index. The input file for this command is an intermediate binary file (BRN) that is produced with the RUNOFF command and the /INTERMEDIATE qualifier (see the RUNOFF command). For a complete description of the DSR indexing utility, see the *VAX DIGITAL Standard Runoff Reference Manual.*

FORMAT

RUNOFF/INDEX *file-spec[,...] or file-spec[+...]*

PARAMETER

file-spec[,...] or file-spec[+...]

Specifies one or more intermediate binary files (BRN) that contain information (index entries, page number references, and so on) for making an index. To create a BRN file, use the RUNOFF command with the /INTERMEDIATE qualifier. See the RUNOFF command for more information on the /INTERMEDIATE qualifier.

If you omit the input file type, the DSR indexing utility uses a default file type of BRN. RUNOFF/INDEX also processes BIX files that the previous version of DSR produced.

For single input files, the indexing utility produces an output file with the same file name as the input file. The output file type is RNX.

If you separate multiple input files with commas, separate RNX files for each input file are created. If you separate multiple input files with plus signs (+), a single RNX file that contains indexing information for all of the input files is created. The default output file name is the same as the first input file name; the default file type is RNX. Wildcard characters are not allowed in the file specification.

DESCRIPTION

The RUNOFF/INDEX command creates an RNX file that can be processed by DSR to produce an index. The formatted index is a 2-column index with balanced columns on each page. This index can be used for draft copies or for final production. Qualifiers to this command allow you to specify the following characteristics for index entries:

- Running page numbers or chapter-oriented page numbers
- The number of lines of index entries per page
- Special text and heading on the first page of the index

QUALIFIERS

/IDENTIFICATION
/NOIDENTIFICATION (default)

Reports the current version number of the DSR indexing utility.

/LINES_PER_PAGE=n

The value n specifies the number of lines of index entries on each page of the finished index. This number does not include the number of lines required for headings and footings.

The default is 55 lines. This value is designed to work properly in the default formatting environment of DSR. You must calculate the value n if you change the default environment in any of the following ways:

- If you use subtitles in the document that requires the RNX file
- If you make the page length for the document anything other than 58 lines per page
- If you use any .LAYOUT other than zero (0)

To calculate the correct value for /LINES_PER_PAGE use the following formula:

```
/LINES_PER_PAGE=n
n = .PAGE SIZE ( the first parameter is length value)
    minus 4 if subtitles are used, minus 3 if no subtitles
    minus the number of lines reserved for .LAYOUT 1,
          .LAYOUT 2, or .LAYOUT 3.
```

/LOG
/NOLOG (default)

Controls whether the DSR index utility displays the name of each input file as it is processed and after it is processed, as well as the name of each output file created. If there are any errors in processing, INDEX sends messages to the terminal even if /NOLOG is in effect.

/OUTPUT[=file-spec]
/NOOUTPUT

Specifies that an output file is to be produced and optionally names it. If you specify the /OUTPUT qualifier without a file specification, or if you omit the qualifier entirely, the output file name matches the input file name. The default file type is RNX.

You can change the name of the output file by supplying a file specification for the value file-spec.

The /NOOUTPUT qualifier suppresses the creation of an output file. You can use /NOOUTPUT to check an input file for errors without using system resources to generate an output file.

/PAGE_NUMBERS=option

Controls whether the page number references in the index are running page numbers or chapter-oriented page numbers. To specify the type of page numbers you want, select from the following options:

Option	Purpose
NORUNNING	Specifies chapter-oriented page numbers (such as 1-3, 10-42). You can specify chapter-oriented numbers for an index even if they do not appear in the document. NORUNNING is the default.
RUNNING	Specifies running page numbers (such as 1, 50, 230). You can specify running page numbers for an index even if the document does not display running page numbers.

/REQUIRE=file-spec
/NOREQUIRE (default)

Allows you to change the heading on the first page of an index. The default heading is the word INDEX centered on the page and followed by three blank lines. The substitute heading is contained in the file you specify, which can contain DSR commands and text.

To change the heading:

1. Create or edit a file that specifies the format and the text that you want as the heading on the first index page.
2. Use the file you create as the file-spec for /REQUIRE.

When you use the /REQUIRE qualifier, the default heading for the first page of the index is not generated. Your file must provide the heading. The file can contain DSR commands and text that you want to appear at the top of the first page of the index, or it can contain only DSR commands. For example, you can put the DSR command .FIGURE 10 in the file. This command generates 10 lines of white space at the top of the first page of the index. You can use these blank lines for later pasteup. See the *VAX DIGITAL Standard Runoff Reference Manual* for a sample file that changes the index heading.

If you are adding lines of text or white space to the heading on the first page of the index, you must allow space for this addition. Use the /RESERVE=n qualifier to provide the space you need.

See the /RESERVE qualifier for more information.

/RESERVE=n
/NORESERVE (default)

Allows you to reserve space at the top of the first page of the index for text or white space that you want to include with the /REQUIRE=file-spec qualifier. Determine how many lines of text or white space you are adding to the top of the first page of the index. Use this number as the value n for the /RESERVE qualifier.

EXAMPLES

1 `$ RUNOFF/INTERMEDIATE CHPT1,CHPT2,CHPT3`

Before using RUNOFF/INDEX, you must create a BRN file as input for the DSR indexing utility. The command in this example creates three separate files: CHPT1.BRN, CHPT2.BRN, and CHPT3.BRN.

```
2  $ RUNOFF/INDEX CHPT1.BRN
```

In this example, the RUNOFF/INDEX command takes the file CHPT1.BRN as input and creates CHPT1.RNX, which can be processed by DSR to produce an index for Chapter 1.

```
3  $ RUNOFF/INDEX/LINE_PER_PAGE=52 CHPT2
```

In this example, the RUNOFF/INDEX command takes the file CHPT2.BRN as input and creates CHPT2.RNX. The RNX file produces an index with 52 lines of index entries per page. The lines per page had to be adjusted because the writer used a page layout with the page numbers centered at the bottom of the page (.LAYOUT 1, .LAYOUT 2, .LAYOUT 3). This page layout takes up three more spaces than .LAYOUT 0, which is the default for DSR. To produce the final index, you must use the RNX file as input to DSR. See the following example.

```
4  $ RUNOFF CHPT2.RNX
```

In this example, the RUNOFF command produces CHPT2.MEX, which is a formatted index. You can type or print this file to view the index.

SEARCH

Searches one or more files for the specified string(s) and displays those lines containing that string or strings.

FORMAT

SEARCH *file-spec[,...] search-string[,...]*

PARAMETERS

file-spec[,...]

Specifies one or more files to be searched. You must specify at least one file name. If you specify two or more file names, separate them with commas.

Wildcard characters are allowed in the file specification.

search-string[,...]

Specifies the character string to be located in the specified files. Enclose strings containing lowercase letters, blanks, or other nonalphanumeric characters (including spaces) in quotation marks.

You can use the /MATCH and /EXACT qualifiers to alter the way that SEARCH matches search strings.

DESCRIPTION

The SEARCH command searches through files for specific character strings; all lines containing occurrences of the strings are displayed. Use the SEARCH qualifiers to tailor the search operation to your specific needs.

The SEARCH command opens the file with shared read and write access. Therefore, any file that has its attributes set to shared write is searched even if it is currently opened by other users.

QUALIFIERS

/BACKUP

Modifies the time value specified with the /BEFORE or /SINCE qualifier. /BACKUP selects files according to the dates of their most recent backups. This qualifier is incompatible with the other qualifiers that also allow you to select files according to time attributes: /CREATED, /EXPIRED, and /MODIFIED. If you specify none of these four time qualifiers, the default is /CREATED.

/BEFORE[=time]

Selects only those files dated prior to the specified time. You can specify time as an absolute time, as a combination of absolute and delta times, or as one of the following keywords: TODAY (default), TOMORROW, or YESTERDAY. Specify one of the following qualifiers with /BEFORE to indicate the time attribute to be used as the basis for selection: /BACKUP, /CREATED (default), /EXPIRED, or /MODIFIED.

See the *VMS DCL Concepts Manual* for complete information on specifying time values.

/BY_OWNER[=uic]

Selects only those files whose owner user identification code (UIC) matches the specified owner UIC. The default UIC is that of the current process.

Specify the UIC using standard UIC format as described in Section 8.1 of the *VMS DCL Concepts Manual*.

/CONFIRM
/NOCONFIRM (default)

Controls whether a request is issued before each SEARCH operation to confirm that the operation should be performed on that file. The following responses are valid:

YES	NO	QUIT
TRUE	FALSE	CTRL/Z
1	0	ALL
	RET	

You can use any combination of uppercase and lowercase letters for word responses. Word responses can be abbreviated to one or more letters (for example, T, TR, or TRU for TRUE), but these abbreviations must be unique. Affirmative answers are YES, TRUE, and 1. Negative answers are NO, FALSE, 0, and <RET>. QUIT or CTRL/Z indicates that you want to stop processing the command at that point. When you respond with ALL, the command continues to process, but no further prompts are given. If you type a response other than one of those in the list, DCL issues an error message and redisplays the prompt.

/CREATED (default)

Modifies the time value specified with the /BEFORE or /SINCE qualifier. /CREATED selects files based on their dates of creation. This qualifier is incompatible with the other qualifiers that also allow you to select files according to time attributes: /BACKUP, /EXPIRED, and /MODIFIED. If you specify none of these four time qualifiers, the default is /CREATED.

/EXACT
/NOEXACT (default)

Controls whether the SEARCH command matches the search string exactly or treats uppercase and lowercase letters as equivalents. By default, SEARCH ignores case differences in letters.

Specifying the /EXACT qualifier causes the system to use less CPU time. Therefore, if you are sure of the case of the letters in the string, it is more efficient to use /EXACT.

/EXCLUDE=(file-spec[,...])

Excludes the specified files from the SEARCH operation. You can include a directory but not a device in the file specification. Wildcard characters are allowed in the file specification. However, you cannot use relative version numbers to exclude a specific version. If you provide only one file specification, you can omit the parentheses.

/EXPIRED

Modifies the time value specified with the /BEFORE or /SINCE qualifier. /EXPIRED selects files according to their expiration dates. (The expiration date is set with the SET FILE/EXPIRATION_DATE command.) The

/EXPIRED qualifier is incompatible with the other qualifiers that also allow you to select files according to time attributes: /BACKUP, /CREATED, and /MODIFIED. If you specify none of these four time qualifiers, the default is /CREATED.

/FORMAT=option

Formats output in one of five ways:

DUMP	Displays all control characters (including <HT> , <CR> , and <LF>) and nonprintable characters as ANSI mnemonics.
NONULLS	Same as DUMP, but removes all null characters from the input file before reformatting. (In DUMP mode, the null character is displayed as <NUL> .) NONULLS is convenient when you are searching binary format files, such as EXE or OBJ files, that generally contain many zero bytes.
NOFF	Replaces control characters in text with ANSI mnemonics (for example, CTRL/C is replaced with <ETX>). The terminal formatting characters <HT> , <CR> , <LF> , <VT> are passed without change. Form feed characters are replaced with <FF> .
PASSALL	Moves control and nonprintable characters to the output device without translating them. The terminal driver cannot send 8-bit characters to the terminal unless either SET TERMINAL/PASSALL or SET TERMINAL/EIGHT_BIT is already in effect. You can use /FORMAT=PASSALL whenever you do not want the SEARCH command to substitute the ANSI mnemonic for control characters (for example, <BEL> for CTRL/G).
TEXT	Replaces control characters in text with ANSI mnemonics (for example, CTRL/C is replaced with <ETX>). The terminal formatting characters <HT> , <CR> , <LF> , <VT> , and <FF> are passed without change. TEXT is the default format.

/HEADING (default)
/NOHEADING

Includes file names in the output file and displays a line of 30 asterisks as a window separator between groups of lines that belong to different files. With the default heading format, file names are printed only when more than one file is specified or when wildcard characters are used.

The /WINDOW qualifier displays a line of 15 asterisks separates each window within a file.

/LOG
/NOLOG (default)

Outputs a message to the current SYS$OUTPUT device for each file searched. The message includes the file name, the number of records, and the number of matches for each file searched.

/MATCH=option

Interprets and matches multiple search strings in one of the following ways:

AND	A match occurs only if the record contains all the strings.
NOR	A match occurs only if the record contains none of the strings.
NAND	A match occurs only if the record does not contain all of the strings.
OR	A match occurs if the record contains any of the strings.

When only one search string is specified, the OR and AND options produce identical results. Similarly, NOR and NAND produce identical results for a single search string.

/MODIFIED

Modifies the time value specified with the /BEFORE or /SINCE qualifier. /MODIFIED selects files according to the dates on which they were last modified. This qualifier is incompatible with the other qualifiers that also allow you to select files according to time attributes: /BACKUP, /CREATED, and /EXPIRED. If you specify none of these four time modifiers, the default is /CREATED.

/NUMBERS
/NONUMBERS (default)

Controls whether the source line number is displayed at the left margin of each line in the output.

/OUTPUT[=file-spec]
/NOOUTPUT

Controls whether the results of the search are output to a specified file. The output is sent to the current default output device (SYS$OUTPUT) if you omit the /OUTPUT qualifier or omit the file specification with the qualifier. The /NOOUTPUT qualifier means that no matching records are output as a result of the SEARCH command.

/REMAINING
/NOREMAINING (default)

Includes in the output all records from the first matched record to the end of the file. This qualifier overrides the value n in /WINDOW, but allows /WINDOW=n1.

/SINCE[=time]

Selects only those files dated after the specified time. You can specify time as an absolute time, a combination of absolute and delta times, or as one of the following keywords: TODAY (default), TOMORROW, or YESTERDAY. Specify one of the following qualifiers with /BEFORE to indicate the time attribute to be used as the basis for selection: /BACKUP, /CREATED (default), /EXPIRED, or /MODIFIED.

See the *VMS DCL Concepts Manual* for complete information on specifying time values.

/STATISTICS
/NOSTATISTICS (default)

Controls whether the following statistics about the search are displayed:

- Number of files searched
- Number of records searched

- Number of characters searched
- Number of records matched
- Number of lines printed
- Buffered I/O count
- Direct I/O count
- Number of page faults
- Elapsed CPU time
- Elapsed time

/WINDOW[=(n1,n2)]
/NOWINDOW (default)

Specifies the number of lines to be displayed with the search string.

If you specify the /WINDOW qualifier without the value n1 and n2, two lines above the search string, the search string, and the two lines below the search string are included in the output. If you specify /WINDOW with a single number (n1), n1 specifies the number of lines to display including the search string. Half the lines precede the matched search string and half follow it. (If n is even, 1 line is added to the lines following the matched search string.)

For example, if you specify /WINDOW=10, nine additional lines are listed along with the line containing the search string. Four lines are listed above the line containing the search string and five lines are listed below it, for a total of ten lines.

If you specify n1 and n2, the /WINDOW qualifier displays n1 lines above the search string, the search string, and n2 lines below the search string. Either of these numbers can be zero.

If you specify /WINDOW=0, the file name of each file containing a match (but no records) is included in the output. This specification creates a file (using the /OUTPUT qualifier) that can be inserted into a command file to manipulate the files containing matches.

If you omit the /WINDOW qualifier, only the line containing a match is displayed.

EXAMPLES

1
```
$ SEARCH CABLE.MEM,JOYNER.MEM "MANUAL TITLE"
```

This command searches the files CABLE.MEM and JOYNER.MEM for occurrences of the character string MANUAL TITLE. Each line containing the string is displayed at the terminal. It is necessary to enclose the string in quotation marks because it contains a space character.

2
```
$ SEARCH/OUTPUT=RESULTS.DAT/WINDOW=9 DISLIST.MEM NAME
```

The SEARCH command searches the file DISLIST.MEM for occurrences of the character string NAME and sends the output to the file RESULTS.DAT. The four lines preceding and following each occurrence of NAME are included in the output.

```
3  $ SEARCH/OUTPUT=ALLSUB.COM/WINDOW=5000 *.COM SUBMIT
```

The SEARCH command searches all command files in the current directory for the string SUBMIT. If a match is found, SEARCH effectively copies the entire command file to the output file, because the window is so large.

```
4  $ SEARCH/OUTPUT=COLUMBUS.OH/WINDOW=(3,0)/NOHEAD/MATCH=AND -
   $_ *.DAT COLUMBUS,OH
```

The SEARCH command searches all files of type DAT for lines containing both COLUMBUS and OH. When a match is found, the three previous lines (containing blank line, name, and street address) are copied to the new file. The new file COLUMBUS.OH is ready to use, because it does not contain headings and window separators.

```
5  $ SEARCH/OUTPUT=SWAP.LIS/FORMAT=PASSALL/NUMBERS/EXACT -
   $_ /WINDOW=10000 SWAP.PAS SWAP
```

This SEARCH command produces a listing file with the line numbers at the left margin. The /FORMAT=PASSALL qualifier is specified so that form-feed characters in the source are passed through. The /EXACT qualifier is specified for efficiency (because it is known that the name SWAP in the program statement is always in uppercase). The /WINDOW qualifier is entered so that the entire file is copied to the output file SWAP.LIS.

```
6  $ SEARCH/REMAINING CABLE.LOG FORTRAN
```

The SEARCH command displays all the lines in the CABLE.LOG file that follow the first occurrence of the string FORTRAN.

```
7  $ SEARCH OMAHA::DISK1:[EXP]SUB.DAT,DATA.LIS VAX
```

The SEARCH command searches through the files SUB.DAT and DATA.LIS at remote node OMAHA for all occurrences of the string VAX. The list of all records containing the string VAX is displayed at the local terminal.

SET

Defines or changes, for the current terminal session or batch job, characteristics associated with files and devices owned by the process.

FORMAT

SET *option*

DESCRIPTION

The SET command options are described individually in this manual. Table DCL–13 lists all the SET command options, including those generally reserved for use by system operators and managers.

Table DCL–13 SET Command Options

Option	Function
ACCOUNTING	Initializes the accounting log file
ACL	Associates an access control list with one or more system objects
AUDIT	Enables or disables forms of security auditing
BROADCAST	Determines which messages will be broadcast to SYS$OUTPUT
CARD_READER	Defines the default ASCII translation mode for a card reader
CLUSTER/QUORUM	Sets the cluster quorum to a value that you specify, or if no value is specified, sets the cluster quorum to a value determined by the system
COMMAND	Adds commands that are defined in a command description file to your process command set or a command tables file
[NO]CONTROL	Enables or disables interrupts caused by CTRL/T or CTRL/Y
DAY	Overrides the default day type specified in the user authorization file (UAF)
DEFAULT	Establishes a device and directory as the current default for file specifications
DEVICE	Defines device characteristics
DEVICE/SERVED	Lets you make a disk on a local node available to all the nodes on a VAXcluster
DIRECTORY	Modifies the characteristics of one or more directories
FILE	Modifies the characteristics of one or more files
FILE/ACL	Modifies the access control list of one or more files
HOST	Connects your terminal to a remote VAX processor by way of the current host processor

Table DCL–13 (Cont.) SET Command Options

Option	Function
HOST/DTE	Connects your system to a remote system by way of an outgoing terminal line
KEY	Changes the current keypad state setting
LOGINS	Allows or disallows users to log in to the system
MAGTAPE	Defines characteristics of a magnetic tape device
MESSAGE	Overrides or supplements system messages
[NO]ON	Controls whether the command interpreter checks for an error condition following the execution of commands in a command procedure
OUTPUT_RATE	Sets the rate at which output is written to a batch job log file
PASSWORD	Lets users change their own passwords; lets system managers change the system password
PRINTER	Defines printer characteristics
PROCESS	Defines execution characteristics of the current process
PROMPT	Defines the DCL prompt
PROTECTION	Defines the protection status of a file or group of files
PROTECTION/DEFAULT	Establishes the default protection to be applied to all files subsequently created
PROTECTION/DEVICE	Establishes the protection to be applied to a specific non-file-structured device
QUEUE	Changes the current status or attributes of the specified queue
QUEUE/ENTRY	Changes the current status or attributes of a job that is not currently executing in a queue
RESTART_VALUE	Establishes a test value for restarting portions of batch jobs
RIGHTS_LIST	Lets users modify the process rights list; lets privileged users modify the system rights list
RMS_DEFAULT	Provides default multiblock and multibuffer count values to be used by RMS for file operations
SYMBOL	Controls access to local and global symbols in command procedures
TERMINAL	Defines terminal characteristics
TIME	Resets the system clock to the specified value
UIC	Changes the UIC of the current process
[NO]VERIFY	Controls whether the command interpreter displays lines in command procedures as it executes them
VOLUME	Modifies the characteristics of one or more Files–11 volumes
WORKING_SET	Changes the current working set limit or quota

SET ACCOUNTING

Enables or disables the logging of various activities in the accounting log file SYS$MANAGER:ACCOUNTNG.DAT. You can also use SET ACCOUNTING to close the current accounting log file and open a new one with a version number incremented by 1.

Requires OPER privilege.

FORMAT

SET ACCOUNTING

PARAMETERS

None.

DESCRIPTION

Disables or enables the logging of the specified activities recorded in the accounting log file. If you specify only /DISABLE, the logging of all activities is disabled. If you specify only /ENABLE, the logging of all activities is enabled.

For a detailed description of the accounting log file records, see the *VMS Accounting Utility Manual*.

QUALIFIERS

/DISABLE[=(keyword[,...])]

Disables the logging of all activities in the accounting log file. To disable specific activities, you include one or more keywords with /DISABLE. When you specify more than one keyword, separate keywords with commas and enclose the entire list in parentheses. You can specify the following keywords:

Keyword	Function
BATCH	Inhibits/allows the recording of batch job termination
DETACHED	Inhibits/allows the recording of detached process termination
IMAGE	Inhibits/allows the recording of image activation
INTERACTIVE	Inhibits/allows the recording of interactive job termination
LOGIN_FAILURE	Inhibits/allows the recording of login failures
MESSAGE	Inhibits/allows the recording of user messages
NETWORK	Inhibits/allows the recording of network job termination
PRINT	Inhibits/allows the recording of all print jobs
PROCESS	Inhibits/allows the recording of all process termination
SUBPROCESS	Inhibits/allows the recording of all subprocess termination

/ENABLE[=(keyword[,...])]

Enables the logging of all activities in the accounting file. To enable specific activities, you include one or more keywords with /ENABLE. Use the same keywords with /ENABLE that you use with /DISABLE. When you specify more than one keyword, separate keywords with commas and enclose the entire list within parentheses.

/NEW_FILE

Closes the current accounting file and opens a new version of that file.

EXAMPLES

1 `$ SET ACCOUNTING/ENABLE=(BATCH,INTERACTIVE)`

The command in this example requests that all batch and interactive jobs be recorded in the accounting file at job termination.

2 `$ SET ACCOUNTING/NEW_FILE`

The command in this example closes the current accounting file and creates a new version of it.

SET ACL

Allows you to create or modify the access control list (ACL) of an object.

FORMAT

SET ACL *object-name*

PARAMETER

object-name

Specifies the object whose access control list (ACL) is being modified. Wildcard characters are allowed in object names only if the object type is FILE. Each file must be a disk file on a Files–11 Structure Level 2 formatted volume. Logical name tables must be system logical name tables.

DESCRIPTION

The SET ACL command enables you to manipulate an entire access control list (ACL) of an object, or to create, modify, or delete access control entries (ACEs) in the ACL of an object. (For information on the format of ACEs and ACLs, see the *VMS Access Control List Editor Manual*.) To use the SET ACL command, specify the object name of the object whose ACL you want to manipulate.

The SET ACL command is used to add ACEs to an ACL by specifying the ACEs with the /ACL qualifier. For example, the following command adds an ACE to the ACL of the file SALARY85.DAT so that all users associated with the identifier PERSONNEL are allowed READ access to the file:

```
$ SET ACL/ACL=(IDENTIFIER=PERSONNEL,ACCESS=READ) SALARY85.DAT
```

If the object specified with the SET ACL command does not have an ACL, one is created.

The SET ACL command provides the following qualifiers to manipulate ACEs and ACLs in various ways:

/AFTER
/DELETE
/LIKE
/NEW
/REPLACE

You can delete ACEs from an ACL by including the /DELETE qualifier and specifying the ACEs with /ACL. To delete all the ACEs (except those with the PROTECTED option), include the /DELETE qualifier and specify the /ACL qualifier without specifying any ACEs.

You can copy an ACL from one object to another by using the /LIKE qualifier. When using the /LIKE qualifier, you must specify the object type and object name. The ACL of the object specified with /LIKE replaces the ACL of the object given with the command.

You can replace existing ACEs in the ACL of the object specified with the command by using the /REPLACE qualifier. Any ACEs specified with /ACL are deleted and replaced by those specified with /REPLACE.

The /NEW qualifier is used to delete all ACEs (except those with the PROTECTED option) before adding any ACEs specified by /ACL, /LIKE, or /REPLACE.

When referring to existing ACEs with /DELETE, /REPLACE, or /AFTER, the existing ACE may be abbreviated.

By default, any ACEs, except security alarm ACEs, added to an ACL are placed at the top of the ACL. Security alarm ACEs are always positioned at the top of the ACL, regardless of positioning qualifiers. Whenever the system receives a request for access to an object that has an ACL, the system searches each entry in the ACL from the first to the last for the first match it can find. If another match occurs further down in the ACL, it has no effect. Because the position of an ACE in an ACL is so important, you can use the /AFTER qualifier to correctly position an ACE. When you use the /AFTER qualifier, any additional ACEs are added after the specified ACE.

The SET ACL command can also be used with the /EDIT qualifier to invoke the ACL editor. When the /EDIT qualifier is specified, only one object name is allowed. The following qualifiers can be used only when the /EDIT qualifier has been specified.

/JOURNAL
/MODE
/RECOVER

For more information on these qualifiers and the ACL editor in general, see the *VMS Access Control List Editor Manual*.

QUALIFIERS

/ACL[=(ace[,...])]

Specifies one or more access control entries (ACEs) to be modified. When no ACE is specified, the entire access control list is affected. Separate multiple ACEs with commas. The specified ACEs are inserted at the top of the ACL unless the /AFTER qualifier is given. (Note that security alarm ACEs are always placed at the beginning of the ACL.)

/AFTER=ace

Indicates that all access control entries (ACEs) specified with the /ACL qualifier will be added after the ACE specified with the /AFTER qualifier. By default, any ACEs added to the ACL are always placed at the top of the list. (Note that security alarm ACEs are always placed at the beginning of the ACL.)

This qualifier cannot be used with the /EDIT qualifier.

/BEFORE[=time]

Selects only those files dated prior to the specified time. You can specify time as an absolute time, as a combination of absolute and delta times, or as one of the following keywords: TODAY (default), TOMORROW, or YESTERDAY.

See Section 1.4 of the *VMS DCL Concepts Manual* for complete information on specifying time values.

This qualifier cannot be used with the /EDIT qualifier and can be used only with an object that is a file.

/BY_OWNER[=uic]

Selects only those files whose owner user identification code (UIC) matches the specified owner UIC. The default UIC is that of the current process.

Specify the UIC using standard UIC format as described in Section 8.1 of the *VMS DCL Concepts Manual*.

This qualifier cannot be used with the /EDIT qualifier and can be used only with an object that is a file.

/CONFIRM
/NOCONFIRM (default)

Issues a request for confirmation before each modification. The following responses are valid:

YES	NO	QUIT
TRUE	FALSE	CTRL/Z
1	0	ALL
	RET	

You can use any combination of uppercase and lowercase letters for word responses. Word responses can be abbreviated to one or more letters (for example, T, TR, or TRU for TRUE), but these abbreviations must be unique. Affirmative answers are YES, TRUE, and 1. Negative answers are NO, FALSE, 0, and the RETURN key. QUIT or CTRL/Z indicates that you want to stop processing the command at that point. When you respond with ALL, the command continues to process, but no further prompts are given. If you type a response other than one of those in the list, DCL issues an error message and redisplays the prompt.

/CREATED

Modifies the ACLs of files selected according to their creation date. Relevant only with the /BEFORE and /SINCE qualifiers.

This qualifier cannot be used with the /EDIT qualifier.

/DEFAULT

Creates an ACL for the specified files as if the files were newly created. For a directory file, the /DEFAULT qualifier propagates the entire ACL (except ACEs with the NOPROPAGATE option) so that a particular access protection can be propagated throughout a directory tree. For all other files, the /DEFAULT qualifier propagates the DEFAULT option ACEs in the ACL of the parent directory to the ACL of the specified files.

The /DEFAULT qualifier uses the ACL of the parent directory of the specified file, not the current default directory.

This qualifier cannot be used with the /EDIT qualifier and can be used only with an object that is a file.

/DELETE

Indicates that the access control entries (ACEs) specified with the /ACL qualifier are to be deleted. If no ACEs are specified with /ACL, the entire ACL is deleted (except those with the PROTECTED option). If you specify an ACE that was not specified with the /ACL qualifier, you are notified that the ACE does not exist, and the delete operation continues.

/EDIT

Invokes the ACL Editor and allows you to use the /JOURNAL, /MODE, or /RECOVER qualifiers. Any other qualifiers specified with /EDIT are ignored.

For more information on the ACL Editor, see the *VMS Access Control List Editor Manual*.

/EXCLUDE=(file-spec[,...])

Excludes the specified files from the SET ACL operation. You can include a directory but not a device in the file specification. Wildcard characters are allowed in the file specification. However, you cannot use relative version numbers to exclude a specific version. If you provide only one file specification, you can omit the parentheses.

This qualifier cannot be used with the /EDIT qualifier and can be used only with an object that is a file.

/JOURNAL[=file-spec]
/NOJOURNAL

Controls whether a journal file is created from the editing session. By default, a journal file is created if the editing session ends abnormally.

If you omit the file specification, the journal file has the same name as the input file and a file type of JOU. You can use the /JOURNAL qualifier to specify a journal file name that is different from the default. No wildcard characters are allowed in the /JOURNAL file-spec parameter.

You must specify /EDIT in order to use this qualifier.

/LIKE=(OBJECT_TYPE=type,OBJECT_NAME=name)

Deletes the ACL of the specified object and replaces it with the ACL of the object specified with /LIKE. Any existing ACEs (except those with the PROTECTED option) are deleted before the ACL specified by /LIKE is copied. ACEs with the NOPROPAGATE option are not copied.

You can specify the following keywords for OBJECT_TYPE: DEVICE, FILE, SYSTEM_GLOBAL_SECTION, GROUP_GLOBAL_SECTION, QUEUE, or LOGICAL_NAME_TABLE. The object-name is specified as it is specified for the command. No wildcard characters are allowed in the /LIKE parameters.

This qualifier cannot be used with the /EDIT qualifier.

/LOG
/NOLOG (default)

Controls whether the SET ACL command displays the object name of the object that has been affected by the command.

This qualifier cannot be used with the /EDIT qualifier.

/MODE=[NO]PROMPT

Determines whether the ACL editor prompts for field values. By default, the ACL editor selects prompt mode.

You must specify the /EDIT qualifier to use this qualifier.

/NEW

Indicates that any existing ACE in the ACL of the object specified with SET ACL (except those with the PROTECTED option) is to be deleted. To use the /NEW qualifier, you must specify a new ACL or ACE with the /ACL, /LIKE, or /REPLACE qualifier.

This qualifier cannot be used with the /EDIT qualifier.

/OBJECT_TYPE=type

Specifies the type of the object whose ACL is being edited. By default, the ACL editor assumes that the object whose ACL is being edited is a file. If the object is not a file, the /OBJECT qualifier is required. Possible keywords are as follows: FILE (includes directory files), DEVICE, SYSTEM_GLOBAL_SECTION, GROUP_GLOBAL_SECTION, QUEUE, or LOGICAL_NAME_TABLE.

/RECOVER[=file-spec]
/NORECOVER (default)

Specifies the name of the journal file to be used in a recovery operation. If the file specification is omitted with /RECOVER, the journal is assumed to have the same name as the input file and a file type of JOU. No wildcard characters are allowed with the /RECOVER file-spec parameter.

You must specify /EDIT in order to use this qualifier.

/REPLACE=(ace[,...])

Deletes the access control entries (ACEs) specified with the /ACL qualifier and replaces them with those specified with /REPLACE. Any ACEs specified with the /ACL qualifier must exist and must be specified in the order in which they appear in the ACL.

This qualifier cannot be used with the /EDIT qualifier.

/SINCE[=time]

Selects only those files dated after the specified time. You can specify time as an absolute time, a combination of absolute and delta times, or as one of the following keywords: TODAY (default), TOMORROW, or YESTERDAY.

See Section 1.4 of the *VMS DCL Concepts Manual* for complete information on specifying time values.

This qualifier cannot be used with the /EDIT qualifier and can be used only with an object that is a file.

EXAMPLES

1
```
$ SET QUEUE/PROTECTION=WORLD LN03_PRINT
$ SET ACL/OBJECT_TYPE=QUEUE/ACL=(IDENTIFIER=SECRETARIES,ACCESS=WRITE) -
_$ LN03_PRINT
```

This example shows how you can use ACLs to limit access to specific queues on the system. (By default, all users can submit jobs to any queues on the system.) The first command in the example removes world WRITE access to the LN03_PRINT print queue, prohibiting all users from submitting jobs to the queue. The second command adds an ACL to the queue allowing WRITE access only to users who hold the SECRETARIES identifier.

2
```
$ SET ACL/LIKE=(OBJECT_TYPE=FILE,OBJECT_NAME=USER.LIS) ACCOUNTS.LIS
```

This example replaces the ACL of the file ACCOUNTS.LIS with the ACL for the file USER.LIS.

SET AUDIT

Enables or disables security auditing on a VMS system. The SET AUDIT command can also be used to specify the auditing failure mode on the system. (Note that you must specify the /ALARM qualifier when enabling or disabling security auditing.)

Requires the SECURITY privilege.

FORMAT

SET AUDIT

PARAMETERS

None.

DESCRIPTION

The SET AUDIT command enables security auditing to send alarms to terminals that have been enabled as security operators whenever specified events are detected by the system. For example, you can use a SET AUDIT command to cause an alarm message to occur if break-in attempts are observed or if the system UAF or network proxy UAF file is modified.

For details on how to enable a terminal for security alarms, see the REPLY /ENABLE command.

The /FAILURE_MODE qualifier specifies the action VMS takes when depleted resources prevent a security alarm from being written. By default, VMS places the process in the MWAIT (miscellaneous wait) state to wait for the resource (/FAILURE_MODE=WAIT). Optionally, you can specify /FAILURE_MODE=IGNORE to ignore failed security alarms or /FAILURE_MODE=CRASH to force a system failure if security alarms cannot be written.

To display the results of a SET AUDIT command, enter the DCL command SHOW AUDIT. To display only the audit failure mode setting, enter the command SHOW AUDIT/FAILURE_MODE.

Because the security auditing features entail a certain amount of system overhead, you should be careful in selecting the features that will provide the most benefit in your work environment. In particular, be aware that enabling alarms for all events can result in a large number of alarm messages being sent to a security terminal. This situation affects the use of the security terminal, because alarm messages have priority over any other I/O from a terminal.

QUALIFIERS

/ALARM

Causes alarm messages to be sent to all terminals enabled as security operators. See the description of the DCL command REPLY/ENABLE for details on how to enable terminals as security operators. The /ALARM qualifier is required when enabling or disabling security either /ENABLE or /DISABLE are required.

/DISABLE=(keyword[,...])

Disables security auditing for the specified events. To disable alarms for all events, specify the keyword ALL. You can also specify the appropriate keywords to selectively disable alarms for from one to all events that are currently enabled. You must specify at least one keyword. See the /ENABLE qualifier description for a list of the keywords to use with the /DISABLE qualifier.

In processing the SET AUDIT command, the system processes the /DISABLE qualifier last. If you accidentally specify both /ENABLE and /DISABLE in the same command, the /DISABLE qualifier prevails.

/ENABLE=(keyword[,...])

Enables security auditing for the specified events. To enable alarms for all events, specify the keyword ALL. You can also specify the appropriate keywords to selectively enable alarms for from one to all events that are currently enabled. You must specify at least one keyword.

The possible events that may be specified in the keyword list of either the /ENABLE or /DISABLE qualifier are as follows:

ACL	An event requested by an access control list (ACL) item, including ACLs on files and global sections.
ALL	All possible events.
AUDIT	An event resulting from the execution of a SET AUDIT command.
AUTHORIZATION	The modification of any portion of the system user authorization file (SYSUAF) or network proxy authorization file (NETPROXY), including any password changes; the modification of any portion of the rights database.
BREAKIN=(keyword[,...])	The occurrence of one or more of the following classes of break-in attempts, as specified by one or more of the keywords:

ALL	All possible sources of break-ins, as defined by the remaining keywords
DETACHED	Detached process break-in attempt
DIALUP	Dialup break-in attempt
LOCAL	Local break-in attempt
NETWORK	Network server break-in attempt
REMOTE	Remote break-in attempt

FILE_ACCESS=(keyword[,...])

The occurrence of file and global section access events (regardless of the value specified in the file's access control list, if any). You can specify one or more of the following keywords to describe the file access event to be noted.

ALL	All types of file access events, as defined by the remaining keywords.
BYPASS [:access [,access...]]	Successful file access due to the use of the BYPASS privilege
FAILURE [:access [,access...]]	Unsuccessful file access
GRPPRV [:access [,access...]]	Successful file access due to the use of the GRPPRV privilege
READALL [:access [,access...]]	Successful file access due to the use of the READALL privilege
SUCCESS [:access [,access...]]	Successful file access
SYSPRV [:access [,access...]]	Successful file access due to the use of the SYSPRV privilege

Most of the keywords permit you to define the type of file access that was obtained with the following keywords:

ALL	All types of file access events, as defined by the remaining keywords. If no access types are specified, ALL is assumed by the system.
READ	READ access
WRITE	WRITE access
EXECUTE	EXECUTE access
DELETE	DELETE access
CONTROL	Owner access

INSTALL

The occurrence of any INSTALL operations.

LOGFAILURE=(keyword[,...])

The occurrence of one or more of the following classes of login failure, as specified by one or more of the keywords:

ALL	All possible types of login failures, as defined by the remaining keywords
BATCH	Batch process login failure
DETACHED	Detached process login failure
DIALUP	Dialup interactive login failure
LOCAL	Local interactive login failure
NETWORK	Network server task login failure
REMOTE	Interactive login failure from another network node, for example, with a SET HOST command
SUBPROCESS	Subprocess login failure

LOGIN=(keyword[,...])

The occurrence of one or more of the following classes of login attempts, as specified by one or more of the keywords:

ALL	All possible sources of logins, as defined by the remaining keywords
BATCH	Batch process login
DETACHED	Detached process login
DIALUP	Dialup interactive login
LOCAL	Local interactive login
NETWORK	Network server task login
REMOTE	Interactive login from another network node, for example, with a SET HOST command
SUBPROCESS	Subprocess login

LOGOUT=(keyword[,...])	The occurrence of one or more of the following classes of logouts, as specified by one or more of the keywords:
	ALL — All possible sources of logouts, as defined by the remaining keywords
	BATCH — Batch process logout
	DETACHED — Detached process logout
	DIALUP — Dialup interactive process logout
	LOCAL — Local interactive process logout
	NETWORK — Logout by a network server task
	PROCESS — Subprocess or detached process logout
	REMOTE — Logout of a process that logged in interactively from another network node
MOUNT	The issuing of a MOUNT or DISMOUNT request

/FAILURE_MODE

Specifies how VMS proceeds following a failed attempt to write a security alarm.

Option	Description
WAIT	Indicates that processes are placed in the MWAIT state to wait until the resource is available. This is the default.
IGNORE	Indicates that failing security alarms are to be ignored. The first failed alarm causes an error message to be written to the operator console and log file. The system maintains a count of the lost alarms, which can be displayed with SHOW AUDIT.
CRASH	Forces a system failure if security alarms cannot be written.

EXAMPLES

1

```
$ SET AUDIT/ALARM/ENABLE=ALL
$ SHOW AUDIT
Security alarm failure mode is set to:
    WAIT        Processes will wait for resource

Security alarms currently enabled for:
    ACL
    MOUNT
    AUTHORIZATION
    BREAKIN:      (DIALUP,LOCAL,REMOTE,NETWORK,DETACHED)
    LOGIN:        (BATCH,DIALUP,LOCAL,REMOTE,NETWORK,SUBPROCESS,DETACHED)
    LOGFAILURE:   (BATCH,DIALUP,LOCAL,REMOTE,NETWORK,SUBPROCESS,DETACHED)
    LOGOUT:       (BATCH,DIALUP,LOCAL,REMOTE,NETWORK,SUBPROCESS,DETACHED)
    FILE_ACCESS:
        FAILURE: (READ,WRITE,EXECUTE,DELETE,CONTROL)
        SUCCESS: (READ,WRITE,EXECUTE,DELETE,CONTROL)
        SYSPRV:  (READ,WRITE,EXECUTE,DELETE,CONTROL)
        BYPASS:  (READ,WRITE,EXECUTE,DELETE,CONTROL)
        GRPPRV:  (READ,WRITE,EXECUTE,DELETE,CONTROL)
        READALL: (READ,WRITE,EXECUTE,DELETE,CONTROL)
```

The SET AUDIT command in this example enables alarms for all possible events; the display from the SHOW AUDIT command identifies the possible events. Note that, by default, VMS places processes in the MWAIT state to wait for resources if an attempt to write a security alarm fails.

2

```
$ SET AUDIT/ALARM/ENABLE=(AUTHORIZATION,BREAKIN)
```

The SET AUDIT command in this example enables alarms at all terminals established as security operators for any change in the system user or network proxy authorization file and for any break-in attempts.

3

```
$ SET AUDIT/FAILURE_MODE=IGNORE
```

The SET AUDIT command in this example directs the VMS operating system to ignore security alarms if they cannot be written and to maintain a count of the lost security alarms. Enter the command SHOW AUDIT to display the number of lost alarm messages.

4

```
$ SET AUDIT/ALARM/DISABLE=ALL
```

The SET AUDIT command in this example disables all alarms to security operators.

SET BROADCAST

Enables you to selectively screen out various kinds of messages from being broadcast to your terminal.

FORMAT

SET BROADCAST=*(class-name[,...])*

PARAMETER

class-name

Specifies the class of message that you want to enable or disable for broadcast to your terminal. If you specify only one class, you can omit the parentheses. The class names are as follows:

ALL	All message classes enabled
[NO]DCL	CTRL/T and SPAWN/NOTIFY messages
[NO]GENERAL	All normal REPLY messages or messages from $BRDCST
[NO]MAIL	Notification of mail
NONE	All message classes disabled
[NO]OPCOM	Messages issued by OPCOM
[NO]PHONE	Messages from the Phone Utility
[NO]QUEUE	Messages referring to print or batch jobs issued by the queue manager
[NO]SHUTDOWN	Messages issued from REPLY/SHUTDOWN
[NO]URGENT	Messages issued from REPLY/URGENT
[NO]USER1 - [NO]USER16	Messages from the specified user groups. See the description of the $BRKTHRU system service in the *VMS System Services Reference Manual* for information on setting up user-written broadcast messages.

DESCRIPTION

The SET BROADCAST command enables you to receive certain kinds of messages at your terminal, but not others. By default, you receive all messages at your terminal (SET BROADCAST=ALL). SET BROADCAST=NONE screens out all messages.

After you have used the SET BROADCAST command to screen out some classes of messages (for example, SET BROADCAST=NOPHONE), you can use the command to restore that class (SET BROADCAST=PHONE).

Use the SHOW BROADCAST command to see which message classes are currently being screened out.

Note: SYS$COMMAND must be a terminal.

EXAMPLES

1
```
$ SET BROADCAST=(NOMAIL, NOPHONE)
.
.
.
$ SET BROADCAST=MAIL
```

In this example, the first SET BROADCAST command screens out all mail and phone messages. Later the second SET BROADCAST command restores mail messages. Phone messages are still screened.

2
```
$ SET BROADCAST=NONE
.
.
.
$ SET BROADCAST=(SHUTDOWN, URGENT, DCL, OPCOM)
```

In this example, the first SET BROADCAST command screens out all messages. Later the second SET BROADCAST command restores shutdown, urgent, DCL, and OPCOM messages. General, phone, mail, queue, and user messages are still screened.

SET CARD_READER

Defines the default translation mode for cards read from a card reader. All subsequent input read from the specified card reader are converted using the specified mode.

FORMAT

SET CARD_READER *device-name[:]*

PARAMETER

device-name[:]

Specifies the name of the card reader for which the translation mode is to be set. The device must not be currently allocated to any other user.

DESCRIPTION

When the system is bootstrapped, the translation mode for cards read into all card readers is set at 029. If you do not specify either of the command qualifiers, the SET CARD_READER command has no effect; that is, the current translation mode for the device remains the same.

QUALIFIERS

/026

Sets the card reader for cards punched on an 026 punch.

/029

Sets the card reader for cards punched on an 029 punch.

/LOG
/NOLOG (default)

Controls whether log information is displayed at the terminal to confirm that the card reader is set.

EXAMPLES

1
```
$ ALLOCATE CR:
  _CRA0: ALLOCATED
$ SET CARD_READER CRA0:/029
$ COPY  CRA0: [MALCOLM.DATAFILES]CARDS.DAT
```

The ALLOCATE command requests the allocation of a card reader by specifying the generic device name. When the ALLOCATE command displays the name of the device, the SET CARD_READER command sets the translation mode at 029. Then the COPY command copies all the cards read by the card reader CRA0 into the file CARDS.DAT in the directory [MALCOLM.DATAFILES].

SET CLUSTER/EXPECTED_VOTES

Sets the total expected votes in the cluster to a value that you specify or, if no value is specified, sets the total votes to a value determined by the system.

Requires OPER privilege and the /EXPECTED_VOTES qualifier.

FORMAT

SET CLUSTER/EXPECTED_VOTES *[=value]*

PARAMETERS

None.

DESCRIPTION

The SET CLUSTER/EXPECTED_VOTES command enables you to adjust the total number of expected votes in the cluster. Set this value equal to the number of votes contributed by each node in the cluster plus the number of votes contributed by the cluster quorum disk. The system will automatically calculate the value of the cluster quorum from the total number of expected votes in the cluster.

You can specify the expected total votes value as part of the SET CLUSTER /EXPECTED_VOTES command string. If you enter the SET CLUSTER /EXPECTED_VOTES command without specifying a value for expected votes, the system calculates the value for you, using the following formula:

```
EXPECTED_VOTES = (NODE1_VOTES + NODE2_VOTES + ...) + QUORUM_DISK_VOTES
```

where NODEn_VOTES is the value of the SYSGEN parameter VOTES for each node in the cluster and QUORUM_DISK_VOTES is the value of the SYSGEN parameters QDSKVOTES.

When you enter the SET CLUSTER/EXPECTED_VOTES command without specifying a value, the system assumes that all nodes that are expected to be in the cluster are currently members.

In general, you use the SET CLUSTER/EXPECTED_VOTES command only when a node is leaving the cluster for an extended period of time. Under normal circumstances, quorum is not reduced when a node leaves the cluster, because it is assumed that the node may be rebooted and rejoin the cluster. If a node is removed from the cluster and is unable to rejoin the cluster within a reasonable period of time (for example, if a node crashes due to a hardware problem and cannot rejoin the cluster for several days), the quorum for the cluster can safely be reduced by lowering the total expected votes until that node rejoins.

The purpose cluster quorum is to eliminate any possibility of the cluster partitioning into separate clusters and simultaneously accessing the same resources (such as HSC50 disks). If the sum of the votes of all members of the cluster is smaller than the cluster quorum, all nodes in the cluster will block activity until new nodes join to increase the vote total. Lowering the quorum value (by reducing the value of the total expected votes) when one or more nodes leave the cluster for long periods of time reduces this possibility.

SET CLUSTER/EXPECTED_VOTES

When you enter the SET CLUSTER/EXPECTED_VOTES command, either with or without an expected votes value specified, the system responds with a message indicating the new value that was actually set. Note that you need only enter this command on one node in the cluster, because the new value for total expected votes is propogated through the cluster. This new expected votes value should then be stored in the SYSGEN parameter EXPECTED_VOTES on each cluster node, so that it remains in effect after the nodes reboot.

When a node that was previously a member of the cluster is ready to rejoin, you should increase the SYSGEN parameter EXPECTED_VOTES to its original value before bringing the node back to the cluster. Note that you do not need to use the SET CLUSTER/EXPECTED_VOTES command to increase the number of expected votes, because the expected votes value will be increased automatically when the node rejoins the cluster.

EXAMPLES

1 `$ SET CLUSTER/EXPECTED_VOTES`

The SET CLUSTER command in this example instructs the system to calculate the total expected votes value for you, because no value is specified as part of the command string. The system uses the NODEn_VOTES + QUORUM_DISK_VOTES formula.

2 `$ SET CLUSTER/EXPECTED_VOTES=9`

The SET CLUSTER command in this example sets the total expected votes to 9, which is the value specified in the command string.

SET COMMAND

Invokes the Command Definition Utility to add commands to your process command table or to a specified command table file. For a complete description of the Command Definition Utility, including information about the SET COMMAND command, see the *VMS Command Definition Utility Manual*.

FORMAT

SET COMMAND *[file-spec[,...]]*

SET CONTROL

Enables or disables CTRL/Y or CTRL/T. CTRL/Y interrupts a command and returns you to the DCL command level. CTRL/T momentarily interrupts a command to print a line of statistics.

SET CONTROL=T requires that SET TERMINAL/BROADCAST be set for the information to be displayed at your terminal.

FORMAT

SET [NO]CONTROL*[=(T,Y)]*

PARAMETER

(T,Y)

Specifies that T (CTRL/T) or Y (CTRL/Y) be enabled or disabled. If you specify both characters, separate them with a comma and enclose the list in parentheses. If you do not specify either T or Y, Y is the default.

DESCRIPTION

The CTRL/Y function provides a general-purpose escape from the current operation. CTRL/Y can generally be used during an interactive terminal session to interrupt the current command, command procedure, or program image.

The SET NOCONTROL=Y command can be used for special application programs. When the SET NOCONTROL=Y command is executed in a system-specified command procedure for a particular user at login, that user can communicate only with the application program that controls the terminal.

When SET NOCONTROL=Y is in effect, the INTERRUPT message is displayed, but no interruption takes place.

SET NOCONTROL=Y also disables the CTRL/C cancel function for all commands and programs that do not have special action routines responding to CTRL/C.

The CTRL/T function displays a single line of statistical information about the current process. When you use CTRL/T during an interactive terminal session, it momentarily interrupts the current command, command procedure, or image to display statistics. The statistical information includes the node and user names, the current time, the current process, CPU usage, number of page faults, level of I/O activity, and memory usage. For example:

```
BOSTON::SMITH  16:21:04 EDT       CPU=00:00:03.33 PF=778 IO=296 MEM=277
```

When SET NOCONTROL=T (the default) is in effect, CTRL/T does not cause any statistics to be displayed.

EXAMPLES

1
```
$ SET NOCONTROL=Y
```

The SET CONTROL command in this example disables the CTRL/Y function as well as most CTRL/C functions.

2
```
$ SET CONTROL=T
```

The SET CONTROL command in this example enables the CTRL/T function.

3
```
$ SET NOCONTROL=(T,Y)
```

The SET CONTROL command in this example disables both the CTRL/T and CTRL/Y functions.

4
```
$ CTRL/T
NODE22::SMITH  16:21:04  (DCL)   CPU=00:03:29.39 PF=14802 IO=18652 MEM=68
$ SET NOCONTROL=T
$ CTRL/T
```

As shown in this example, when you press CTRL/T, the system displays the appropriate information. The SET NOCONTROL=T command disables the CTRL/T function. Now when you press CTRL/T, no information is displayed.

SET CLUSTER/QUORUM

Sets the quorum in the cluster to a value that you specify or, if no value is specified, sets the cluster quorum to a value determined by the system. The /QUORUM qualifier is required.

As of VMS Version 5.0, the SET CLUSTER/QUORUM command is superseded by the SET CLUSTER/EXPECTED_VOTES command. DIGITAL recommends the use of the SET CLUSTER/EXPECTED_VOTES command. See SET CLUSTER/EXPECTED_VOTES for a complete description of this command.

Requires OPER privilege.

FORMAT

SET CLUSTER/QUORUM *[=quorum-value]*

SET DAY

Sets the default day type specified in the user authorization file (UAF) for the current day.

Requires OPER privilege.

FORMAT

SET DAY

PARAMETERS

None.

DESCRIPTION

The SET DAY command overrides the default primary and secondary day types that are used to control daily logins. These default day types are defined by the system manager in the user authorization file (UAF).

The SET DAY command is useful when you need to override the day type because of a change in the work days of a particular week. For example, Monday, which is normally a work day, is occasionally a holiday. You can use the SET DAY command to override the normal day type for Monday and set it to a different day type.

QUALIFIERS

/DEFAULT

Overrides any previous SET DAY specification and specifies that the normal UAF defaults are to be used to determine today's day type.

/LOG
/NOLOG (default)

Controls whether log information is displayed at the terminal to confirm that the new SET DAY information has been set.

/PRIMARY

Sets today until midnight to a primary day.

/SECONDARY

Sets today until midnight to a secondary day.

EXAMPLES

1
```
$ SET DAY/PRIMARY
```

The SET DAY command in this example overrides the current default day type and sets the today until midnight to a primary day.

2
```
$ SET DAY/DEFAULT
```

The SET DAY command in this example overrides the previous SET DAY command and sets today's day type to the UAF-defined default.

SET DEFAULT

Sets your default device and directory specifications. The new default is applied to all subsequent file specifications that do not explicitly include a device or directory name.

When you change the default device assignment, the system equates the specified device with the logical name SYS$DISK.

FORMAT

SET DEFAULT *[device-name[:]][directory-spec]*

PARAMETER

device-name[:]

The name of the device you want to go to.

directory-spec

The name of the directory you want to go to. A directory name must be enclosed in brackets. Use the minus sign to specify the next higher directory from the current default.

You must specify either the device-name parameter or the directory-spec parameter. If you specify only the device name, the current directory is the default for the directory-spec parameter. If you specify only the directory name, the current device is the default for the device-name parameter.

You can use a logical name but it must constitute at least the device part of the specification. When you use a search list logical name as the parameter, the logical name is not translated by the SET DEFAULT command. Instead, the SET DEFAULT command retains the logical name so that RMS is able to access the entire search list. If you enter the SHOW DEFAULT command, the search list logical name is displayed as the default device, along with its equivalence strings.

EXAMPLES

1
```
$ SET DEFAULT [CARPENTER]
```

The SET DEFAULT command in this example changes the default directory to [CARPENTER]. The default disk device does not change.

2
```
$ SET DEFAULT $FLOPPY1:[WATER.MEMOS]
```

The SET DEFAULT command in this example sets your default to the WATER.MEMOS subdirectory on $FLOPPY1.

3
```
$ SET DEFAULT $FLOPPY1:
```

The SET DEFAULT command in this example sets the default device to $FLOPPY1. The directory name does not change.

4
```
$ SET DEFAULT [-]
```

The SET DEFAULT command in this example changes the default directory to the parent directory of the one you are currently in. For example, if the current directory is $FLOPPY1:[WATER.MEMOS], this command sets your default to $FLOPPY1:[WATER]. If you are in $FLOPPY1:[WATER], this command sets your default to the master directory on the disk — $FLOPPY1:[000000].

5
```
$ SAVEDEF = F$ENVIRONMENT("DEFAULT")
$ SET DEFAULT [122001.JONES.APP10]
    .
    .
    .
$ SET DEFAULT  'SAVEDEF'
```

The command procedure in this example uses the F$ENVIRONMENT lexical function to save the current default directory in the symbol named SAVEDEF. The SET DEFAULT command changes the default directory 122001.JONES.APP10. Later, the symbol SAVEDEF is used to restore the original default directory.

6
```
$ SHOW DEFAULT
$ DEFINE X WORK:[TOP.SUB1],WORK:[TOP.SUB2]
$ SET DEFAULT X
$ SHOW DEFAULT
X:[TOP]
 =   WORK:[TOP.SUB1]
 =   WORK:[TOP.SUB2]
$ DIRECTORY

Directory WORK:[TOP.SUB1]

FOO.TMP;1

Total of 1 file.

Directory WORK:[TOP.SUB2]

FOO.TMP;1

Total of 1 file.

Grand total of 2 directories, 2 files.
$ DIRECTORY []

Directory WORK:[TOP]

FOO.TMP;1           NETSERVER.LOG;2

Total of 2 files.
```

In this example, the default directory is WORK:[TOP]. X is then defined to be a search list consisting of two subdirectories. When the SET DEFAULT X command is entered, the search list (X) is equated with the logical name SYS$DISK and entered into the disk field. The subsequent SHOW DEFAULT command shows both the search list and the current default directory, followed by the expanded search list.

If a DIRECTORY command is entered, the directories searched are those contained in the logical name X. However, if the current default directory specification ([]) is explicitly entered, the current default directory, rather than SYS$DISK, is searched.

SET DEVICE

Establishes a print device or terminal as a spooled device or establishes the status of error-logging for a device.

Requires OPER privilege.

FORMAT

SET DEVICE *device-name[:]*

PARAMETER

device-name[:]

Specifies the name of the device whose spooling or error-logging status is to change. The device must be a print device or a terminal if its spooling status is to change; the device must be a disk or magnetic tape if its error-logging status is to change.

DESCRIPTION

When you specify the /SPOOLED qualifier, program output that uses VMS RMS or FCS and specifies the print device name is written onto an intermediate disk rather than written directly to the print device or terminal.

When you specify the /ERROR_LOGGING qualifier, all error messages reported by the device on which error-logging is enabled are recorded in the error log file.

QUALIFIERS

/AVAILABLE
/NOAVAILABLE

Controls whether the specified disk is to be considered available. This command can be entered only after the specified disk has been dismounted. If you specify /NOAVAILABLE, any attempt to mount the specified disk is prevented.

/DUAL_PORT
/NODUAL_PORT

Controls whether the port seize logic in the device driver of the specified disk is to be enabled. This qualifier should be used only on disks that contain a dual port kit and have been dismounted.

/ERROR_LOGGING
/NOERROR_LOGGING

Controls whether device errors are logged in the error log file. Use the SHOW DEVICE/FULL command to find out the current status.

/LOG
/NOLOG (default)

Controls whether log information is displayed at the terminal.

/SPOOLED[=(queue-name[:],intermediate-disk-name[:])]
/NOSPOOLED

Controls whether files are spooled to an intermediate disk.

The queue name indicates the printer queue to which a file is queued. If a queue name is not supplied, the default is the name of either the printer or terminal.

The intermediate disk name identifies the disk to which the spooled files are written. If the intermediate disk name is not supplied, the default is SYS$DISK (the current default disk). The intermediate disk must be mounted before files can be written to it.

Once the device has been set spooled to a disk, that disk cannot be dismounted until the spooled device is set to /NOSPOOLED. All channels must be deassigned from a print device before its spooling characteristics can be changed. Also, the queue that is assigned to the device must be stopped.

EXAMPLES

1 `$ SET DEVICE/SPOOLED=(LPA0)  LPA0:`

In this example, the /SPOOLED qualifier requests that the printer queue LPA0 be spooled to an intermediate disk before files directed to the disk are printed. Because no intermediate disk was specified, the intermediate disk defaults to SYS$DISK.

2 `$ SET DEVICE/ERROR_LOGGING  DBB2:`

The SET DEVICE command in this example requests that all device errors reported on DBB2 be logged in the error log file.

3 `$ SET DEVICE/NOAVAILABLE DRA0:`

The SET DEVICE command in this example prevents any attempt to mount a disk on DRA0.

4 `$ SET DEVICE/DUAL_PORT DRA0:`

The SET DEVICE command in this example enables the dual port seize logic in DRA0.

SET DEVICE/SERVED

Allows you to make a disk on a local node available to all the nodes in a cluster. The /SERVED qualifier is required.

Applies only to VAXcluster environments.

FORMAT

SET DEVICE/SERVED *node-name$DDcu:*

PARAMETER

node-name$DDcu:
Specifies the device name of the device that you want to make available to the cluster.

DESCRIPTION

The SET DEVICE/SERVED command is used in conjunction with the Mass Storage Control Protocol (MSCP) server to make a disk on a local node available to all nodes on the cluster. The local node must be a member of a VAXCluster, and the local MSCP server must have been invoked by the SYSGEN utility.

Note: Unless the disk device that you intend to make available to the cluster is a system disk, it must not already be mounted when you enter the SET DEVICE/SERVED command.

The SET DEVICE/SERVED command string can be included as part of the local startup command file, and entered before the MOUNT utility mounts the disk to be served (made available to the entire cluster).

EXAMPLE

```
$ SET DEVICE/SERVED DRA4:
```

The SET DEVICE/SERVED command in this example instructs the MSCP server to make the disk device DRA4: on your local node available to all other processors on your cluster.

SET DIRECTORY

Modifies the characteristics of one or more directories.

See the qualifier descriptions for restrictions.

FORMAT

SET DIRECTORY *[device-name[:]]directory-spec[,...]*

PARAMETER

device-name[:]

Specifies the device on which the directory that you want to modify is located. The device name parameter is optional.

directory-spec[,...]

Specifies one or more directories to be modified. If you specify two or more directories, separate them with commas. Wildcard characters are allowed.

QUALIFIERS

/BACKUP

Modifies the time value specified with the /BEFORE or /SINCE qualifier. /BACKUP selects files according to the dates of their most recent backups. This qualifier is incompatible with the other qualifiers that also allow you to select files according to time attributes: /CREATED, /EXPIRED, and /MODIFIED. If you specify none of these four time qualifiers, the default is /CREATED.

/BEFORE[=time]

Selects only those files dated prior to the specified time. You can specify time as an absolute time, as a combination of absolute and delta times, or as one of the following keywords: TODAY (default), TOMORROW, or YESTERDAY. Specify one of the following qualifiers with /BEFORE to indicate the time attribute to be used as the basis for selection: /BACKUP, /CREATED (default), /EXPIRED, or /MODIFIED.

See Section 1.4 of the *VMS DCL Concepts Manual* for complete information on specifying time values.

/BY_OWNER[=uic]

Selects only those files whose owner user identification code (UIC) matches the specified owner UIC. The default UIC is that of the current process.

Specify the UIC using standard UIC format as described in Section 8.1 of the *VMS DCL Concepts Manual.*

/CONFIRM
/NOCONFIRM (default)

Controls whether a request is issued before each SET DIRECTORY operation to confirm that the operation should be performed on that file. The following responses are valid:

YES	NO	QUIT
TRUE	FALSE	CTRL/Z
1	0	ALL
	RET	

You can use any combination of uppercase and lowercase letters for word responses. Word responses can be abbreviated to one or more letters (for example, T, TR, or TRU for TRUE), but these abbreviations must be unique. Affirmative answers are YES, TRUE, and 1. Negative answers are NO, FALSE, 0, and RETURN. QUIT or CTRL/Z indicates that you want to stop processing the command at that point. When you respond with ALL, the command continues to process, but no further prompts are given. If you type a response other than one of those in the list, DCL issues an error message and redisplays the prompt.

/CREATED (default)

Modifies the time value specified with the /BEFORE or /SINCE qualifier. /CREATED selects files based on their dates of creation. This qualifier is incompatible with the other qualifiers that also allow you to select files according to time attributes: /BACKUP, /EXPIRED, and /MODIFIED. If you specify none of these four time qualifiers, the default is /CREATED.

/EXCLUDE=(file-spec[,...])

Excludes the specified files from the SET DIRECTORY operation. You can include a directory but not a device in the file specification. Wildcard characters are allowed in the file specification. However, you cannot use relative version numbers to exclude a specific version. If you provide only one file specification, you can omit the parentheses.

/EXPIRED

Modifies the time value specified with the /BEFORE or /SINCE qualifier. /EXPIRED selects files according to their expiration dates. (The expiration date is set with the SET FILE/EXPIRATION_DATE command.) The /EXPIRED qualifier is incompatible with the other qualifiers that also allow you to select files according to time attributes: /BACKUP, /CREATED, and /MODIFIED. If you specify none of these four time qualifiers, the default is /CREATED.

/LOG
/NOLOG (default)

Controls whether the system displays the directory specification of each directory that is modified as the command executes.

/MODIFIED

Modifies the time value specified with the /BEFORE or /SINCE qualifier. /MODIFIED selects files according to the dates on which they were last modified. This qualifier is incompatible with the other qualifiers that also allow you to select files according to time attributes: /BACKUP, /CREATED, and /EXPIRED. If you specify none of these four time modifiers, the default is /CREATED.

/OWNER_UIC[=uic]

Requires SYSPRV privilege to specifiy a UIC other than your own.

Specifies an owner UIC for the directory. The default UIC is that of the current process.

/SINCE[=time]

Selects only those files dated after the specified time. You can specify time as an absolute time, a combination of absolute and delta times, or as one of the following keywords: TODAY (default), TOMORROW, or YESTERDAY. Specify one of the following qualifiers with /BEFORE to indicate the time attribute to be used as the basis for selection: /BACKUP, /CREATED (default), /EXPIRED, or /MODIFIED.

See Section 1.4 of the *VMS DCL Concepts Manual* for complete information on specifying time values.

/VERSION_LIMIT[=n]

Specifies the total number of versions that a file in the specified directory can have. If you do not specify a version limit, a value of 0 is used, indicating that the number of versions of a file is limited only to the Files–11 architectural limit—32,767. If you change the version limit for the directory, the new value applies only to files created after the change has been made.

The SET DIRECTORY version limit value refers to the number of files with the same file name and type that can exist in the directory at one time. It has no effect on the version number field of a particular file specification. Use the SET FILE command to set limits on file version numbers.

To find out the current version limit for a directory, you must use the DUMP /HEADER command. Specify the /FORMATTED qualifier to format the output and the /BLOCKS=COUNT:0 qualifier to avoid dumping the entire directory contents. For example,

```
DUMP/HEADER/FORMATTED/BLOCKS=COUNT:0 directory-spec
```

EXAMPLES

1

```
$ SET DIRECTORY/VERSION_LIMIT=5/CONFIRM [SMITH...]
```

The SET DIRECTORY command in this example sets a version limit of five for all files in the SMITH directory and all subdirectories of [SMITH]. The /CONFIRM qualifier requests that you confirm whether or not the specified directory should actually be modified. Note that it only affects the files created after the command is entered.

2

```
$ SET DIRECTORY/OWNER_UIC=[360,020] [DAVIDSON],[USERS]
```

The SET DIRECTORY command in this example modifies both the [DAVIDSON] and [USERS] directories, changing their owner UICs. Using the OWNER_UIC qualifier requires SYSPRV (system privilege).

SET ENTRY

Changes the current status or attributes of a job that is not currently executing in a queue.

Requires OPER privilege or EXECUTE (E) access to the specified queue. If you have DELETE (D) access to the specified job, you can alter the attributes for that job.

FORMAT

SET ENTRY *entry-number*

PARAMETER

entry-number

Specifies the entry number of the job you want to change. The job number is displayed at the time of the job's submission.

DESCRIPTION

The SET ENTRY command allows you to change the status or attributes of a job that has been submitted to a printer or batch queue, as long as the job is not currently executing. (You cannot affect individual files within a multi-file job with the SET ENTRY command.)

The qualifiers enable you to specify different attributes or delete attributes. Some qualifiers apply to both batch and print jobs. Others are restricted to either batch jobs or print jobs. The defaults for all the SET ENTRY qualifiers are the attributes and status that the job has before you enter the SET ENTRY command.

The system assigns a unique entry number to each queued print or batch job in the system. The PRINT and SUBMIT commands display the job number when they successfully queue a job for processing. You can enter the SHOW QUEUE command to refresh your memory about a job's entry number. Use the job entry number to specify which job you want to change.

QUALIFIERS

/AFTER=time
/NOAFTER

Requests that the specified job be held until after a specific time. If the specified time has already passed, the job is queued for immediate processing.

You can specify either an absolute time or a combination of absolute and delta times. See Section 1.4 of the *VMS DCL Concepts Manual* for complete information on specifying time values.

/BURST[=keyword]
/NOBURST

Controls whether a burst page is included at the beginning of a print job. A burst page precedes a flag page and contains the same information. However, it is printed over the perforation between the burst page and the flag page. The printing on the perforation makes it easy to separate individual print jobs. When you specify /BURST, you need not specify /FLAG; a flag page automatically follows the burst page. Possible keywords are:

ALL	All printed files contain a burst page.
ONE	The first printed file contains a burst page.

Use the /[NO]BURST qualifier to override the installation-defined defaults that have been set for the printer queue you are using.

/CHARACTERISTICS=(characteristic[,...])
/NOCHARACTERISTICS

Enables you to change the characteristics desired for the job. If you specify only one characteristic, you can omit the parentheses. Codes for characteristics can be either names or values from 0 to 127 and are installation-defined. Use the SHOW QUEUE/CHARACTERISTICS command to see which characteristics have been defined for your system. Use the SHOW QUEUE/FULL command to see which characteristics are available on a particular queue.

When you include the /CHARACTERISTICS qualifier with the SET ENTRY command, all the characteristics you specify must also be specified for the queue that will be executing the job. If not, the job remains pending in the queue until the queue characteristics are changed or you delete the entry with the DELETE/ENTRY command. You need not specify every characteristic of a queue with the SET ENTRY command as long as the ones you specify are a subset of the characteristics set for that queue. The job will also run if no characteristics are specified.

Specification of a characteristic for a queue does not prevent jobs that do not specify that characteristic from being executed.

/CLI=filename

Specifies the name of a command language interpreter (CLI) to use in processing the job. The file name specifies that the CLI be SYS$SYSTEM:filename.EXE. If you do not specify the /CLI qualifier, the job is run by the CLI specified in the user's authorization record, or whatever CLI was specified when the job was originally submitted to the queue.

/COPIES=n

Specifies the number of copies to print. The n parameter can be any number from 1 to 255. When you use the /COPIES qualifier with the SET ENTRY command, the number of copies can apply only to the entire job. You cannot use this qualifier to specify different numbers of copies for individual files within a multi-file job.

/CPUTIME=option

Specifies a CPU time limit for the batch job. Time can be specified as: delta time, 0, NONE, or INFINITE. Both the value 0 and the keyword INFINITE allow unlimited CPU time; the keyword NONE defaults to your UAF value or the limit specified on the queue. You cannot specify more time than permitted by the base queue limits or your own UAF. See Section 1.4 of the *VMS DCL Concepts Manual* for information on specifying time values.

When you need less CPU time than authorized, use the /CPUTIME qualifier to override the base queue value established by the system manager or the value authorized in your user authorization file. Specify 0 or INFINITE to request an infinite amount of time. Specify NONE when you want the CPU time to default to your user authorization file (UAF) value or the limit specified on the queue. Note that you cannot request more time than permitted by the base queue limits or your own UAF.

/FEED
/NOFEED

Controls whether form feeds are inserted into print jobs when the printer nears the end of a page. The number of lines per form can be reset by the /FORM qualifier. You can suppress this automatic form feed (without affecting any of the other carriage control functions that are in place) by using the /NOFEED qualifier.

When you use the /FEED qualifier with the SET ENTRY command, the qualifier applies to all files in the print job. You cannot use this qualifier to specify form feeds for individual files within a multi-file job.

/FLAG[=keyword]
/NOFLAG

Controls whether a flag page is printed preceding a print job. The flag page contains the name of the user submitting the job, the job entry number, and other information about the job. You can specify one of the following keywords:

ALL	Prints a flag page before each file in the job
ONE	Prints a flag page before the first file in the job

Use the /[NO]FLAG qualifier to override the installation-defined defaults that have been set for the printer queue you are using.

/FORM=type

Specifies the name of the form that you want for the print job. Specify the form type using a numeric value or alphanumeric code. Form types can refer to the width, length, or type of paper. Codes for form types are installation-defined. You can use the SHOW QUEUE/FORM command to find out the form types available for your system. The SHOW QUEUE/FULL command tells you which form is set for a specific queue.

If you specify a form type different from that of the queue, your job remains pending until the form type of the queue is set equal to the form type of the job or you delete the job with the DELETE/ENTRY command. You can use the SET ENTRY command to change the form type of your job to match that of the queue. To change the form type for the queue, stop the queue, physically change the form, and restart the queue, specifying the new form type.

/HEADER
/NOHEADER

Controls whether a heading line is printed at the top of each output page in a print job.

/HOLD
/NOHOLD

Controls whether or not the job is to be made available for immediate processing or held for processing later. If you specify /HOLD, the job is not released for processing until you specifically release it with the /NOHOLD or /RELEASE qualifier. You can use the SET ENTRY command to release a job that was previously submitted with a /HOLD qualifier, or you can place a job on hold so that it will run later.

You can use the /NOHOLD qualifier to release jobs that have been held for the following reasons:

- A job was submitted with the /HOLD qualifier.
- A completed job is being held in a queue that has /RETAIN specified.
- A user-written symbiont has refused a job.

/JOB_COUNT=n

Requests that an entire print job be printed n times, where n is a decimal integer from 1 to 255. This qualifier overrides the /JOB_COUNT qualifier specified or defaulted with the PRINT command.

/KEEP
/NOKEEP

Controls whether the batch job log file is deleted after it is printed.

/LOG_FILE=file-spec
/NOLOG_FILE

Creates a log file with the specified file specification. You can specify a different device name, as long as the process executing the batch job has access to the device on which the log file will reside. Logical names in the file specification are translated in the context of the process that executes the SET ENTRY command.

If you omit the /LOG_FILE qualifier and specify the /NAME qualifier, the log file is written to a file having the same file name as that specified by the /NAME qualifier; the file type is LOG. When you omit the /LOG_FILE qualifier, the job-name value used with /NAME must be a valid file name.

/LOWERCASE
/NOLOWERCASE

Indicates whether the files must be printed on a printer that can print both uppercase and lowercase letters. The /NOLOWERCASE qualifier means that files can be printed on printers supporting only uppercase letters. If all available printers can print both uppercase and lowercase letters, you do not need to specify /LOWERCASE.

/NAME=job-name

Defines a name string to identify the job. The name string can have from 1 to 39 characters. The job name is used in the SHOW QUEUE command display. For batch jobs, the job name is also used for the batch job log file. For print jobs, the job name is also used on the flag page of the printed output.

If the /NAME qualifier has not been specified for the job, the name string defaults to the file name of the first, or only, file in the job; the file type is LOG.

/NOCHECKPOINT

For a batch job, erases the value established by the most recently executed SET RESTART_VALUE command. For a print job, clears the stored checkpoint so that the job will restart from the beginning.

/NODELETE

Cancels file deletion for a job that was submitted with the /DELETE qualifier. If no /DELETE qualifier was specified when the job was originally submitted to the queue, you cannot use the SET ENTRY to establish file deletion at a later time. You cannot use the /NODELETE qualifier to specify that individual files in a multi-file job not be deleted.

/NOTE=string

Specifies a message of up to 255 characters to appear on the flag page of the job. Enclose the message in quotation marks if it contains spaces, special characters, or lowercase characters.

/NOTIFY
/NONOTIFY

Controls whether a message is broadcast to any terminal at which you are logged in, notifying you when your job has been completed or aborted.

/OPERATOR=string

Specifies a message string of up to 255 characters to be sent to the operator just before the job begins execution. When the job begins execution, the queue pauses and the message is transmitted to the operator. Enclose the message in quotation marks if it contains spaces, special characters, or lowercase characters.

/PAGES=([l,]u)

Specifies the number of pages to print for the specified job. You can use the /PAGES qualifier to print portions of a long file. When you use the /PAGES qualifier with the SET ENTRY command, the qualifier can only apply to an entire job. You cannot use this qualifier to specify different numbers of pages to be printed for individual files within a multi-file job.

The l (lower) specifier refers to the first page in the group of pages that you want printed for that job. If you omit the l specifier, the printing starts on the first page of the job. The u (upper) specifier refers to the last page of the file that you want printed. You must use two consecutive quotation marks ("") if you omit the upper parameter.

When you want to print to the end of the file but do not know how many pages that will be, you can use two consecutive quotation marks ("") as the u specifier. You can omit the parentheses when you specify only a value for u. For example, /PAGES=10 prints the first 10 pages of the job; /PAGES=(5,10) prints pages 5 through 10; /PAGES=(5,"") starts printing at page 5 and continues until the end of the job is reached.

/PARAMETERS=(parameter[,...])

Specifies from 1 to 8 optional parameters to be passed to the job. Each parameter can have as many as 255 characters. If you specify only one parameter, you can omit the parentheses. The commas delimit individual parameters. To specify a parameter that contains any special characters or delimiters, enclose the parameter in quotation marks.

For batch jobs, the parameters define values to be equated to the symbols named P1 through P8 in each command procedure in the job. The symbols are local to the specified command procedures.

/PASSALL
/NOPASSALL

Specifies whether the symbiont bypasses all formatting and sends the output QIO to the driver with format suppressed. All qualifiers affecting formatting, as well as the /HEADER, /PAGES, and /PAGE_SETUP qualifiers, will be ignored.

When you use the /PASSALL qualifier with the SET ENTRY command, the qualifier applies to the entire job. You cannot use this qualifier to specify PASSALL mode for individual files within a multi-file job.

/PRINTER[=queue-name]
/NOPRINTER

Queues the batch job log to the specified printer queue when the job is completed. By default, the printer queue for the log file is SYS$PRINT. The /PRINTER qualifier allows you to specify a particular printer queue. The /NOPRINTER qualifier assumes the /KEEP qualifier.

/PRIORITY=n

Requires OPER or ALTPRI privilege to raise the priority above the value of the SYSGEN parameter MAXQUEPRI.

Specifies the priority of the job. The priority value must be in the range of 0 through 255, where 0 is the lowest priority and 255 is the highest. The default value for /PRIORITY is the value of the SYSGEN parameter DEFQUEPRI. No privilege is needed to set the priority lower than the MAXQUEPRI value.

/RELEASE

Releases for processing jobs submitted with the /HOLD qualifier or /AFTER qualifier, jobs held in a queue with the /RETAIN qualifier, and jobs refused by a user-written symbiont.

/REQUEUE=queue-name[:]

Requests that the job be moved from the original queue to the specified queue.

/RESTART
/NORESTART

Specifies whether a batch or print job will be restarted after a system crash or a STOP/QUEUE/REQUEUE command.

/SETUP=module[,...]

Extracts the specified module from the device control library (containing escape sequence modules for programmable printers) and copies the module to the printer before a file is printed.

When you use the /SETUP qualifier with the SET ENTRY command, the qualifier applies to the entire job. You cannot use this qualifier to specify different setup modules for individual files within a multi-file job.

/SPACE
/NOSPACE

The /SPACE qualifier specifies that the output is to be double-spaced. Specifying /NOSPACE causes the output to be single-spaced. When you use the /SPACE qualifier with the SET ENTRY command, the qualifier

applies to the entire job. You cannot use this qualifier to specify different spacing for individual files within a multi-file job.

/TRAILER[=keyword]
/NOTRAILER

Controls whether a trailer page is printed at the end of a job. The trailer page displays the job entry number, as well as information about the user submitting the job.

When you use the /TRAILER qualifier with the SET ENTRY command, trailer pages are placed at the end of each file in a multi-file job. Possible keywords are as follows:

ALL	All printed files contain a trailer page
ONE	The last printed file contains a trailer page

Use the /[NO]TRAILER qualifier to override the installation-defined defaults that have been set for the printer queue you are using.

/WSDEFAULT=n

Defines a working set default for a batch job. Possible values are a positive integer in the range 1 through 65,535, 0, or the keyword NONE (the default) for n.

Use this qualifier to override the base queue value established by the system manager or the value authorized in the user authorization file (UAF), provided you want to impose a lower value. Specify 0 or NONE if you want the working set value defaulted to either the UAF value or the working set quota specified on the queue. You cannot request a value higher than the default.

/WSEXTENT=n

Defines a working set extent for a batch job. Possible values are a positive integer in the range 1 through 65,535, 0, or the keyword NONE (the default) for n.

Use this qualifier to override the base queue value established by the system manager or the value authorized in the user authorization file (UAF), provided you want to impose a lower value. Specify 0 or NONE if you want the working set extent defaulted to either the UAF or the working set extent specified on the queue. You cannot request a value higher than the default.

/WSQUOTA=n

Defines the maximum working set size for a batch job. This is the working set quota. Possible values are: a positive integer in the range 1 through 65,535, 0, or the keyword NONE (the default) for n.

Use this qualifier to override the base queue value established by the system manager or the value authorized in the user authorization file (UAF), provided you want to impose a lower value. Specify 0 or NONE if you want the working set quota defaulted to either the user authorization file value or the working set quota specified on the queue. You cannot request a value higher than the default.

EXAMPLES

1

```
$ PRINT/HOLD   MYFILE.DAT
  Job MYFILE (queue SYS$PRINT, entry 112) holding
    .
    .
    .
$ SET ENTRY 112/RELEASE/JOB_COUNT=3
```

The PRINT command in this example requests that the file MYFILE.DAT be queued to the system printer, but placed in a hold status. The SET ENTRY command releases the file for printing and changes the number of copies of the job to three.

2

```
$ SUBMIT CLIMATE
  Job CLIMATE (queue SYS$BATCH, entry 211) pending
$ SET ENTRY 211/HOLD/NAME=TEMP
```

The SUBMIT command in this example queues a command procedure for batch processing. The system assigns a job entry number of 211. The SET ENTRY command places the second job in a hold state and changes the job name to TEMP, assuming that job 211 had not yet begun execution.

3

```
$ PRINT/FLAG=ALL/AFTER=20:00 MEMO.MEM, LETTER.MEM, REPORT.MEM/SPACE
  Job MEMO (queue SYS$PRINT, entry 172) holding until 20:00
    .
    .
    .
$ SET ENTRY 172 /BURST/NOSPACE/HEADER
```

The PRINT command in this example requests that three files be printed after 8:00 P.M. on the default printer with flag pages preceding each file. It also requests that the file REPORT.MEM be double-spaced. Later a SET ENTRY command is issued. This command calls for a burst page at the beginning of each file and requests that all files in the job be single-spaced. Headers are printed on each page of each file in the job.

SET FILE

Modifies the characteristics of one or more files.

See the qualifier descriptions for restrictions.

FORMAT

SET FILE *file-spec[,...]*

PARAMETER

file-spec[,...]

Specifies one or more files to be modified. If you specify two or more files, separate them with commas. Wildcard characters are allowed.

DESCRIPTION

The SET FILE command modifies a number of file characteristics. Use the SET FILE/ACL command to modify the access control list of one or more files.

QUALIFIERS

/ACL

Modifies an access control list (ACL) associated with one or more files. For more information, see the description of the SET FILE/ACL command.

/BACKUP
/NOBACKUP

Specifies that BACKUP records the contents of the file. The /NOBACKUP qualifier causes BACKUP to record the attributes of the file but not its contents. Valid only for Files–11 Structure Level 2 files.

The /NOBACKUP qualifier is useful for saving files that contain unimportant data, such as SWAPFILES.

/BEFORE[=time]

Selects only those files dated prior to the specified time. You can specify time as an absolute time, as a combination of absolute and delta times, or as one of the following keywords: TODAY (default), TOMORROW, or YESTERDAY.

See Section 1.4 of the *VMS DCL Concepts Manual* for complete information on specifying time values.

/BY_OWNER[=uic]

Selects only those files whose owner user identification code (UIC) matches the specified owner UIC. The default UIC is that of the current process.

Specify the UIC using standard UIC format as described in Section 8.1 of the *VMS DCL Concepts Manual*.

/CONFIRM
/NOCONFIRM (default)

Controls whether a request is issued before each SET FILE operation to confirm that the operation should be performed on that file. The following responses are valid:

YES	NO	QUIT
TRUE	FALSE	CTRL/Z
1	0	ALL
	RET	

You can use any combination of upper- and lowercase letters for word responses. Word responses can be abbreviated to one or more letters (for example, T, TR, or TRU for TRUE), but these abbreviations must be unique. Affirmative answers are YES, TRUE, and 1. Negative answers are NO, FALSE, 0, and <RET>. QUIT or CTRL/Z indicates that you want to stop processing the command at that point. When you respond with ALL, the command continues to process, but no further prompts are given. If you type a response other than one of those in the list, DCL issues an error message and redisplays the prompt.

/CREATED

Modifies the time value specified with the /BEFORE or /SINCE qualifier. /CREATED selects files based on their dates of creation.

/DATA_CHECK[=([NO]READ,[NO]WRITE)]

Specifies whether a READ data check (rereading each record), a WRITE data check (reading each record after it is written), or a combination of the two is performed on the file during transfers. By default, a WRITE data check is performed.

/END_OF_FILE

Resets the end-of-file mark to the highest block allocated.

/ENTER=new-file-spec

Use with caution. Assigns an additional name to a single file so that the file has a second name, or alias. However, both the original name and the alias reference the same file. For this reason, take care when deleting files that have aliases. To keep the file but remove one of its names, use the /REMOVE qualifier with SET FILE.

No wildcards are allowed in the file specification.

/ERASE_ON_DELETE

Specifies that the specified files are erased from the disk (not just merely written over) when the DELETE or PURGE command is issued for the files. See DELETE/ERASE for more information.

/EXCLUDE=(file-spec[,...])

Excludes the specified file from the SET FILE operation. You can include a directory name but not a device name in the file specifications. Wildcard characters are supported for file specifications. However, you cannot use relative version numbers to exclude a specific version.

If you specify only one file, you can omit the parentheses.

/EXPIRATION_DATE=date
/NOEXPIRATION_DATE

Requires ownership of the file or access control.

Controls whether an expiration date is assigned to the specified files.

Specify the date according to the rules described in Section 1.4 of the *VMS DCL Concepts Manual*. Absolute date keywords are allowed. If you specify 0 as the date, today's date is used.

/EXTENSION[=n]

Sets the extend quantity default for the file. The value of *n* can range from 0 through 65,535. If you omit the value specification or specify a value of 0, VAX RMS calculates its own /EXTENSION value.

See the SET RMS_DEFAULT command for a description of the /EXTEND_QUANTITY qualifier.

/GLOBAL_BUFFER=n

Sets the VAX RMS global buffer count (the number of buffers that can be shared by processes accessing the file) for the specified files. The value *n* must be an integer in the range 0 through 32,767. A value of 0 disables buffer sharing.

/LOG
/NOLOG (default)

Displays the file specification of each file modified as the command executes.

/MODIFIED

Modifies the time value specified with the /BEFORE or /SINCE qualifier. /MODIFIED selects files according to the dates on which they were last modified. This qualifier is incompatible with /CREATED, which also allows you to select files according to time attributes. If you do not specify /MODIFIED, the default is /CREATED.

/NODIRECTORY

Use with extreme caution. This qualifier removes the directory attributes of a file and allows you to delete the corrupted directory file even if other files are contained in the directory. When you delete a corrupted directory file, the files contained within it are lost.

Use ANALYZE/DISK_STRUCTURE/REPAIR to place the lost files in [SYSLOST]. You can then copy the lost files to a new directory. This qualifier is valid only for the Files–11 Structure Level 2 files. For more information about the Verify Utility, see the *VMS Analyze/Disk_Structure Utility Manual*.

/OWNER_UIC[=uic]

Requires GRPPRV to set the owner to another member of the same group. Requires SYSPRV to set the owner to any UIC outside your group.

Specifies an owner user identification code (UIC) for the file. The default is the UIC of your process.

Specify the UIC using standard UIC format as described in Section 8.1 of the *VMS DCL Concepts Manual*.

/PROTECTION[=(code)]

Cannot be used to change the protection on a file via DECnet.

Enables you to change or reset the protection for one or more of your files. The ownership categories are SYSTEM, OWNER, GROUP, AND WORLD. The access categories are R (read), W (write), E (execute), and D (delete). If you specify /PROTECTION without the ownership and access code, the file protection is set according to the current default protection.

See Section 8.1 of the *VMS DCL Concepts Manual* for more information on specifying protection code.

/REMOVE

Use with caution. This qualifier enables you to remove one of the names of a file that has more than one name, without deleting the file. If you have created an additional name for a file with the /ENTER qualifier of SET FILE, you can use the /REMOVE qualifier to remove either the original name or the alias. The file still exists and can be accessed by whatever name or names remain in effect.

However, if you accidentally remove the name of a file that has only one name, you cannot access that file with most DCL commands; use the ANALYZE/DISK_STRUCTURE utility to retrieve the file.

/SINCE[=time]

Selects only those files dated after the specified time. You can specify time as an absolute time, a combination of absolute and delta times, or as one of the following keywords: TODAY (default), TOMORROW, or YESTERDAY.

See Section 1.4 of the *VMS DCL Concepts Manual* for complete information on specifying time values.

/STATISTICS
/NOSTATISTICS (default)

Enables the gathering of RMS statistics on the specified file. These statistics can subsequently be viewed using the Monitor Utility, which is invoked with the DCL command MONITOR.

/TRUNCATE

Truncates the file at the end of the block containing the end-of-file (EOF) marker, that is, releases allocated but unused blocks of the file.

/UNLOCK

Makes one or more improperly closed files accessible.

/VERSION_LIMIT[=n]

Specifies the maximum number of versions for the specified file. If you do not specify a version limit, a value of 0 is used, indicating that the number of versions of a file is limited only to the Files–11 architectural limit of 32,767. When you exceed that limit, the earliest version of the file is deleted from the directory without notification to the user. For example, if you set the version limit to 3 when there are already five versions of that file in your directory, there will continue to be five versions of the file unless you specifically delete some or purge the directory. Once the number of versions is equal to or less than the current version limit, the version limit is maintained.

EXAMPLES

1 `$ SET FILE/EXPIRATION_DATE=31-DEC-1988:11:00 BATCH.COM;3`

The SET FILE command requests that the expiration date of the file BATCH.COM;3 be set to 11:00 A.M., December 31, 1988.

2 `$ SET FILE/BEFORE=31-DEC/ERASE_ON_DELETE PERSONNEL*.SAL`

This SET FILE command calls for all files that match the file specification PERSONNEL*.SAL and are dated before December 31 of the current year to have their disk locations erased whenever one of them is deleted with commands such as DELETE or PURGE.

3 `$ SET FILE/OWNER_UIC=[360,020]/VERSION_LIMIT=100 MYFILE.DAT`

The SET FILE command modifies the characteristics of the file MYFILE.DAT, changing the owner UIC and assigning a file version limit of 100. You must have system privilege (SYSPRV) to change the owner UIC.

SET HOST

Connects your terminal (through the current host processor) to another processor, called the remote processor. Both processors must be running DECnet. [1]

- **You can use the SET HOST command only if your system is connected by DECnet to another system.**
- **You must have an account on the remote system to log in after the SET HOST command has made the connection.**
- **The SET HOST command requires the network mailbox privilege NETMBX.**

FORMAT

SET HOST *node-name*

PARAMETER

node-name
Specifies the node name of the remote processor to which you will connect.

DESCRIPTION

The SET HOST command connects you to another processor on a network. (The SHOW NETWORK command lists the names of nodes accessible to your node.) Once the connection is made, the remote processor prompts for a user name and password. You must have an account on the remote processor to log in.

Once you have connected to the remote processor and logged in, you can use DCL commands just as you would on your local processor. You can even use the SET HOST command to connect to another remote processor.

Use the LOGOUT command to log off the last processor you have logged in to. If you have connected to and logged in to more than one processor, the LOGOUT command leaves you logged in to the next previous processor.

For example, if your local node is BOSTON, you can use the command SET HOST ALBANY to connect to the node ALBANY. You can then use the command SET HOST AKRON to connect (still through BOSTON and ALBANY) to the node AKRON.

When you use the LOGOUT command, you have logged off (and disconnected from) the processor at node AKRON, but you are still logged in (and connected) to the processor at ALBANY. A second LOGOUT command logs you off ALBANY, and disconnects you from it. A third LOGOUT command logs you off the local processor, BOSTON.

You can also abort operations and return directly to the original host processor, if necessary. Press CTRL/Y at least two times in rapid succession. The following message is displayed:

```
Are you repeating ^Y to abort the remote session?
```

[1] Available under separate license.

If you respond Y or YES, control returns to the original node. Other responses, such as N or NO, do not abort the connection. This technique is useful when you want to exit quickly without entering a series of LOGOUT commands, or when part of the network becomes disconnected and you want to return to the host.

QUALIFIERS

/BUFFER_SIZE=n

Changes the packet size of the protocol message sent between the terminal and the remote processor if a connection to the remote processor is already established. The default buffer size is 1010 bytes; however, *n* can range from 140 bytes to 1024 bytes. *N* is reset to 140 bytes if a value below 140 is specified; a value for *n* above 1024 bytes is reset to 1024.

You can force the host node to write to the terminal in smaller packets, thereby ensuring that write operations to the terminal are displayed at more frequent intervals, by setting *n* to a value just above the minimum of 140 bytes. On slow DECnet links, setting the buffer size to a smaller value may decrease pauses between write operations when large amounts of data are being scrolled to the screen.

/LOG[=file-spec]
/NOLOG (default)

Controls whether a log file of the entire session is kept. If you use /LOG without the file specification, the log information is stored in the file SETHOST.LOG.

/RESTORE
/NORESTORE

Saves current terminal characteristics before a remote terminal session is begun and restores them when the remote session is terminated.

EXAMPLES

1
```
$ SET HOST ITALIC
Username:  BROWN
Password:
    Welcome to VAX/VMS Version 5.0 on node ITALIC
    .
    .
    .
$  LOGOUT
BROWN logged out at 31-DEC-1988 15:04:25.27
%REM-S-END, Control returned to node _CASLON::
```

In this example, the name of the local node is CASLON. This SET HOST command connects the user terminal to the processor at the network node named ITALIC. The remote processor then prompts for user name and password. Use the normal login procedure to log in to the remote processor.

Once you are logged in at a remote node, you can use the SET HOST command to establish communication with another node. After logging into node ITALIC, you could type SET HOST BODONI.

You would again be prompted for a user name and password. If you then supply a valid user name and password, you will be logged in at node BODONI.

Note that when you log out at node BODONI, control is returned to node ITALIC. You must log out from node ITALIC to return to your local node, CASLON.

2

```
$  SET TERMINAL/WIDTH=80
$  SET HOST/RESTORE GENEVA
Username: Jones
Password:
$  SET TERMINAL/WIDTH=132
   .
   .
   .
$  logout
JONES   logged out at 19-APR-1988 11:04:51.45
%REM-S-END, control returned to node _ORACLE
```

This example shows user JONES on node ORACLE log into remote node GENEVA and specify that the original terminal screen width be restored to 80 characters when the remote session is terminated.

SET HOST/DTE

Connects your system to a remote system through an out-going terminal line. Exit from the remote system by typing CTRL/\; that is, type a backslash (\) while holding down the CTRL key.

You must have an account on the remote system in order to log in to that system after the connection is made.

Also requires the ability to assign a channel to the terminal port specified. By default, BYPASS privilege is required but can be changed by setting the device protection for the terminal port.

FORMAT

SET HOST/DTE *terminal-name*

PARAMETER

terminal-name

Specifies the name of an out-going terminal line, which connects your system directly to another system or to a modem.

DESCRIPTION

The SET HOST/DTE command allows you to connect your terminal to another system. Once the connection is made, the remote system prompts for a user name and password. You must have an account on the remote system to log in.

Once you have connected to the remote system and logged in, you can use DCL commands just as you would on your local system.

To log in on lines that expect a break rather than a carriage return, type CTRL/] (that is, type a bracket (]) while holding down the CTRL key) to generate the break.

When connecting directly to another system, the out-going port should be set to NOTYPEAHEAD. This avoids the possibility of an endless loop, wherein noise on the line causes each of the ports to attempt to initiate a login. (Note that the terminal line to which you have connected by way of SET HOST /DTE, must be set to TYPEAHEAD to allow the login.)

QUALIFIERS

/DIAL=(NUMBER:number[,MODEM_TYPE:modem-type])

Allows a modem attached to the out-going terminal line to be autodialed using the autodial protocol of that modem. The NUMBER keyword is the telephone number to be autodialed and is a required parameter.

The MODEM_TYPE: keyword is optional and can be used to specify a modem-type of DF03, DF112 or DMCL. By default, a modem-type of DF03 is assumed. DMCL is any modem that uses the DEC Modem Command Language.

In addition, MODEM_TYPE: may be used to specify a modem-type other than a DF03, DF112 or DMCL. A template is provided for users interested in supporting other modems with autodial capabilities (see SYS$EXAMPLES:DTE_DF03.MAR).

/LOG[=file-spec]
/NOLOG

Controls whether a log file of the entire session is kept. If you do not specify a file, the log information is stored in the file SETHOST.LOG.

When used to log a modem session, the log file contains any noise that occurred on the phone line. For example, typing a file in order to get it recorded in the log file could result in noise being recorded along with the file data. Therefore, the use of the /LOG qualifier is not recommended for the purpose of file transfers.

Asynchronous DECnet is the recommended way to transfer files. For additional information, see the *VMS Networking Manual*.

EXAMPLES

1
```
$ SET HOST/DTE TTA2:/DIAL=NUMBER:5551234
Username:  SMITH
Password:
```

The SET HOST/DTE command in this example connects the user terminal to the out-going terminal line TTA2:, which is attached to a modem (type DF03 by default) set to autodial the phone number 555-1234. The remote processor then prompts for user name and password. Use the normal login procedure to log in to the remote system.

2
```
$ SET HOST/DTE/DIAL=(NUMBER:5551234#,MODEM_TYPE:DF112) TTA2:
Username: SMITH
Password:
```

The SET HOST/DTE command in this example accomplishes the same thing as in the first example, except that it uses the DF112 modem. Note that the number sign (#) is required to activate the autodialer in the DF112.

SET HOST/DUP

Connects your terminal to a storage contoller through the appropriate bus for that controller.

For use only with storage controllers. Requires the DIAGNOSE privilege.

FORMAT

SET HOST/DUP*/SERVER=server-name /TASK=task-name node-name*

PARAMETER

node-name
Specifies the node name of the storage controller.

DESCRIPTION

The SET HOST/DUP command creates a virtual terminal connection and executes a utility or diagnostic program on a storage controller that uses the Diagnostic and Utilities Protocol Standard Dialogue.

Once the connection is established, operations are performed under the control of the utility or diagnostic program.

When the utility or diagnostic program terminates, control is returned to the local system.

To abort or prematurely terminate the connection and return control to the local system, press CTRL/\; that is, type a backslash (\) while holding down the CTRL key.

Further information regarding the operation of the utility and diagnostic programs available on a particular controller may be obtained from the appropriate documentation set for that controller.

Preparing for Use

To use the SET HOST/DUP facility, you must first install FYDRIVER, the Diagnostic and Utilities Protocol (DUP) class driver. To load FYDRIVER, add the following commands to the SYSTARTUP.COM command procedure in the SYS$MANAGER directory.

```
$ RUN SYS$SYSTEM:SYSGEN
CONNECT FYA0/NOADAPTER
```

This operation requires the CMKRNL privilege.

QUALIFIERS

/LOG[=file-spec]
/NOLOG (default)
Controls whether a log file of the entire session is kept. If you use /LOG without the file specification, the log information is stored in the file HSCPAD.LOG.

/SERVER=server-name

Specifies the server name for the target storage controller.

This qualifier is required.

/TASK=task-name

Specifies the utility or diagnostic name to be executed on the target storage controller under direction of the server.

This qualifier is required.

EXAMPLE

```
$ SET HOST/DUP/SERVER=DUP$/TASK=DIRECT BLKHOL
%HSCPAD-I-LOCPROGEXE, Local program executing - type ^\ to exit utility
```

The SET HOST/DUP command in this example connects the user terminal to the utility program called DIRECT executing on a storage controller named BLKHOL under direction of the DUP$ server.

SET HOST/HSC

Connects your terminal to a remote HSC50 disk and tape controller through the Computer Interconnect bus.

Used only with remote HSC50s. Requires the DIAGNOSE privilege.

FORMAT

SET HOST/HSC *node-name*

PARAMETER

node-name
Specifies the node name of the remote HSC50.

DESCRIPTION

The SET HOST/HSC command establishes a connesction to an HSC50 disk and tape controller by way of the Computer Interconnect (CI) bus. (The SHOW CLUSTER command lists the names of HSC50s that are accessible to your node.) No password is required to access the HSC50; however, only DCL SHOW commands are accepted when the HSC50's Secure/Enable switch is in the SECURE position.

Once the connection is made to the HSC50, operations may be performed as if you were attached to the local terminal of the HSC. However, access to ODT (Octal Debugging Tool) and offline diagnostics is not permitted.

Press CTRL/C to obtain a prompt from the HSC50 before entering commands. To exit from the HSC50 and return to the local system, press CTRL/\; that is, type a backslash (\) while holding down the CTRL key.

A description of HSC50 commands and utilities may be obtained from the *HSC User's Guide*.

Preparing for Use

To use the SET HOST/HSC facility, you must first install FYDRIVER, which is the Diagnostic and Utilities Protocol (DUP) driver associated with the CI. To load FYDRIVER, add the following commands to the SYSTARTUP.COM command procedure in the SYS$MANAGER directory.

```
$ RUN SYS$SYSTEM:SYSGEN
CONNECT FYA0/NOADAPTER
```

This operation requires the privilege CMKRNL.

QUALIFIER

/LOG[=file-spec]
/NOLOG (default)
Controls whether a log file of the entire session is kept. If you use /LOG without the file specification, the log information is stored in the file HSCPAD.LOG.

EXAMPLE

```
$ SET HOST/HSC HSC001
%HSCPAD-I-LOCPROGEXE, Local program executing - type ^\ to exit, ^Y for prompt
HSC50>
```

This SET HOST/HSC command connects the user terminal to the HSC named HSC001.

SET KEY

Sets and locks the key definition state for keys defined with the DEFINE /KEY command.

FORMAT

SET KEY

DESCRIPTION

When you define keypad keys using the DEFINE/KEY command, you can assign a specific state name to the key definition. If that state is not set when you press the key, the definition is not processed. Use the SET KEY command to change the current state to the appropriate state.

QUALIFIERS

/LOG (default)
/NOLOG

Controls whether the system displays a message indicating that the key state has been set.

/STATE=state-name
/NOSTATE

Specifies the name of the state. The state name can be any alphanumeric string. If you omit the /STATE qualifier or use /NOSTATE, the current state is left unchanged. The default state is DEFAULT.

EXAMPLE

```
$ SET KEY /STATE=EDITING
```

The SET KEY command in this example sets the key state to the EDITING state. You can now use the key definitions that were defined for the EDITING state.

SET LOGINS

Sets the interactive limit (number of interactive users allowed on the system), or displays the interactive limit and the current number of interactive users.

Requires OPER privilege.

FORMAT SET LOGINS

PARAMETERS *None.*

DESCRIPTION

The SET LOGINS command is not retroactive. All users logged in to the system before you enter the SET LOGINS command are not affected by the command. However, once the limit you set is reached, no more users can log in to the system until someone else logs out. Users with the OPER privilege are not affected by the limit.

If you do not specify a parameter value with the /INTERACTIVE=n qualifier, the SET LOGINS command displays the following information:

```
Login quotas - Interactive limit=x, Current interactive value=y
```

The value x represents the current interactive limit, and the value y represents the number of users currently logged in to the system.

QUALIFIER

/INTERACTIVE[=n]

Establishes the number of interactive users allowed to gain access to the system. If n is specified, the interactive limit is set to n. If n is not specified, the SET LOGINS command displays the current interactive limit and the number of interactive users.

EXAMPLES

1

```
$ SET LOGINS/INTERACTIVE=5
%SET-T-INTSET, login interactive limit=5, current interactive value=3
```

In this example, the SET LOGINS command specifies that only five interactive users can be logged in to the system.

2

```
$ SET LOGINS/INTERACTIVE
%SET-T-INTSET, login interactive limit=15, current interactive value=6
```

When the SET LOGINS command is entered without a parameter, as shown in this exapmle, the /INTERACTIVE qualifier requests that the current status of the login quotas be displayed. The message returned indicates that the maximum number of interactive users allowed on the system is 15 and that the number of interactive users currently logged in is 6. No change is made.

SET MAGTAPE

Defines the default characteristics associated with a specific magnetic tape device for subsequent file operations.

The SET MAGTAPE command is valid for magnetic tape devices mounted with foreign volumes.

FORMAT

SET MAGTAPE *device-name[:]*

PARAMETER

device-name[:]

Specifies the name of the magnetic tape device for which the characteristics are to be set. The device must not be currently allocated to any other user.

QUALIFIERS

/DENSITY=density

Specifies the default density, in bits per inch (bpi), for all write operations on the magnetic tape device when the volume is mounted as a foreign tape or as an unlabeled tape. The density can be specified as 800, 1600, or 6250, if supported by the magnetic tape drive.

/END_OF_FILE

Writes a tape mark at the current position on the magnetic tape volume.

/LOG
/NOLOG

Displays information about the operations performed on the magnetic tape volume.

/LOGSOFT (default)
/NOLOGSOFT

Controls whether soft errors on the specified device are to be logged in the error log file. Soft errors are errors corrected by the hardware without software intervention. This qualifier only affects devices that support hardware error correction, such as the TU78 magnetic tape drive. When used with other devices, this qualifier has no effect.

/REWIND

Requests that the volume on the specified device be rewound to the beginning of the magnetic tape.

/SKIP=option

Requests that the magnetic tape volume be positioned according to any of the following options:

BLOCK:n	Directs the SET MAGTAPE command to skip the specified number of blocks
END_OF_TAPE	Directs the SET MAGTAPE command to position the volume at the end-of-tape mark
FILES:n	Directs the SET MAGTAPE command to skip the specified number of files
RECORD:n	Directs the SET MAGTAPE command to skip the specified number of records

/UNLOAD

Requests that the volume on the specified device be rewound and unloaded.

EXAMPLES

1
```
$ MOUNT MTB1:/FOREIGN
$ SET MAGTAPE MTB1: /DENSITY=800
```

The MOUNT command in this example mounts a foreign tape on the device MTB1. The SET MAGTAPE command defines the density for writing the magnetic tape at 800 bpi.

2
```
$ MOUNT MTA0:/FOREIGN
$ SET MAGTAPE MTA0:/SKIP=FILES:4
```

The MOUNT command in this example mounts a foreign tape on the device MTA0; the SET MAGTAPE command directs the magnetic tape position to skip four files.

SET MESSAGE

Sets the format for system messages or specifies a process level message file. Lets you override or supplement the system messages.

FORMAT

SET MESSAGE *[file-spec]*

PARAMETER

file-spec

Specifies the name of the process level message file. Messages in this file supersede messages for the same conditions in the system message file or in an existing process message file. The file type defaults to EXE. No wildcard characters are allowed. If you do not specify this parameter, the qualifiers apply to the system message file.

DESCRIPTION

The SET MESSAGE command specifies which message fields VMS displays. The message format is as follows:

```
%FACILITY-L-IDENT, text
```

When a process is initially created, the default is to display all four message fields: facility, severity, identification, and text. To control which fields of a message are displayed, specify the appropriate qualifiers. For example, to omit the FACILITY field in message displays, specify SET MESSAGE /NOFACILITY.

By including the SET MESSAGE command in your login command file, you can select specific portions of the messages for your process.

Use the SET MESSAGE command, also, to override or supplement system messages. Whenever any software detects an error and invokes the $GETMSG system service, the message files are searched in the following order: image message sections first, process-permanent message files second, and the system message file last. Thus, with the SET MESSAGE command, you can introduce messages earlier in the searching order; you can either override or supplement the system messages. (Note that the new message definitions affect only your process.)

If a process-permanent message file exists when you specify the SET MESSAGE command with a file specification, the old file is removed and the new file added.

The message definitions you specify must result from a successful compilation with the MESSAGE command. For full details on how to create your own messages with the Message Utility, see the *VMS Message Utility Manual.*

QUALIFIERS

/DELETE

Removes any process permanent message files currently in effect. Do not specify the file-spec parameter with the /DELETE qualifier.

/FACILITY (default)
/NOFACILITY
Formats messages so that the facility name prefix appears.

/IDENTIFICATION (default)
/NOIDENTIFICATION
Formats messages so that the message identification prefix appears.

/SEVERITY (default)
/NOSEVERITY
Formats messages so that the severity level appears.

/TEXT (default)
/NOTEXT
Formats messages so that the message text appears.

EXAMPLES

1
```
$ SET MESSAGE/TEXT/NOFACILITY/NOIDENTIFICATION/NOSEVERITY
$ SHOW DEVICES/MUONTED
unrecognized qualifier - check validity, spelling, and placement
\MUONTED\
```

The SET MESSAGE command in this example formats the error message so that only the text appears.

2
```
$ TYPE XXX
%TYPE-W-OPENIN, error opening DB1:[MALCOLM]XXX.LIS; as input
-RMS-E-FNF, file not found
    .
    .
    .
$  SET MESSAGE/NOIDENTIFICATION
    .
    .
    .
$ TYPE XXX
%TYPE-W, error opening DB1:[MALCOLM]XXX.LIS; as input
-RMS-E, file not found
```

When the first TYPE command is entered in this example, the error messages include all fields. Later, the SET MESSAGE command establishes that the IDENT portion (the abbreviation for the message text) is omitted in future messages. Note the absence of the IDENT component in the two subsequent messages that result from attempting to type a file that does not exist.

3
```
$ SET MESSAGE NEWMSG
```

The SET MESSAGE command in this example specifies that the message text in NEWMSG.EXE supplements the existing system messages.

SET ON

Enables error checking by the command interpreter after the execution of each command in a command procedure. Specify SET NOON to disable error checking.

FORMAT

SET [NO]ON

PARAMETERS

None.

DESCRIPTION

During the execution of command procedures, the command interpreter normally checks the status code returned when a DCL command or program image completes and saves the numeric value of this code in the reserved symbol named $STATUS. The low-order three bits of this value are also saved in the reserved symbol $SEVERITY. Command procedure execution aborts when either an error or fatal error is detected.

Use the SET NOON command to override default error checking. When SET NOON is in effect, the command interpreter continues to place the status code value in $STATUS and the severity level in $SEVERITY, but does not perform any action based on the values. As a result, the command procedure continues to execute no matter how many errors are returned.

The SET ON or SET NOON command applies only at the current command level. If you use the SET NOON command in a command procedure that executes another procedure, the default, SET ON, is established while the second procedure executes.

EXAMPLE

```
$ SET NOON
$ DELETE  *.SAV;*
$ SET ON
$ COPY  *.OBJ  *.SAV
```

This command procedure routinely copies all object modules into new files with the file type SAV. The DELETE command first deletes all existing files with the SAV file type, if any. The SET NOON command ensures that the procedure continues executing even if there are no files with the SAV file type in the current directory. Following the DELETE command, the SET ON command restores error checking. Then the COPY command makes copies of all existing files with OBJ file type.

SET OUTPUT_RATE

Sets the rate at which output is written to a batch job log file.

For use only within command procedures that are submitted as batch jobs.

FORMAT

SET OUTPUT_RATE*[=delta-time]*

PARAMETER

delta-time

The time interval at which output is written from the output buffer to the batch job log file. If no delta time is specified, the information is written in the output buffer to the log file, but the output rate is not changed from the default of once per minute. Specify *delta-time* as [dddd-][hh:mm:ss.cc]. See Section 1.4 of the *VMS DCL Concepts Manual* for more information on delta time.

DESCRIPTION

When you submit a batch job, the output to be written to the log file is stored in an output buffer. Periodically, the buffer is flushed and its contents are written to the batch job log file. By default, the buffer is flushed once a minute. Therefore, you can type the log file to determine how much of the job has completed while the job is still executing.

To change the default output rate, include the SET OUTPUT_RATE command in the command procedure that you are submitting as a batch job. When the SET OUTPUT_RATE command is executed within a batch job and a delta time is specified, DCL flushes the buffer, sets the default output rate, and starts a new output interval.

If the SET OUTPUT_RATE command is executed within a batch job and a delta time is not specified, DCL flushes the buffer but does not change the default output rate and does not start a new interval.

If you issue the SET OUTPUT_RATE command interactively, or within a command procedure that is executed interactively, then no action is performed.

EXAMPLE

```
$ SET OUTPUT_RATE=:0:30
    .
    .
    .
```

This command, when executed within a batch job, changes the default output rate from once a minute to once every 30 seconds.

SET PASSWORD

Establishes, changes, or removes a password. SET PASSWORD can be used by users to change their own passwords and by system managers to change the system password.

See the qualifier descriptions for restrictions.

FORMAT

SET PASSWORD

PARAMETERS

None.

DESCRIPTION

All user accounts on a system have passwords. A password is required for logging in to the system.

To maintain secrecy, users should change their passwords from time to time. The SET PASSWORD command offers a means of making this change.

A system manager can control which users have the right to change their passwords. In addition, a system manager can establish a minimum acceptable password length and the maximum period of time that a password may remain unchanged. When a password has expired, you must use the SET PASSWORD command to change your password.

Systems can also have passwords (not to be confused with the password associated with the SYSTEM account). The system manager uses the SET PASSWORD/SYSTEM command to change the system password from time to time.

A password contains up to 31 alphanumeric characters. The dollar sign ($) and underscore (_) are also permitted. Uppercase and lowercase characters are equivalent. All lowercase characters are converted to uppercase before the password is encrypted. (For example, **EAGLE** is the same as **eagle.**)

Use the following procedure to change your password:

1. Enter the SET PASSWORD command.
2. The system prompts you for your current password. Enter your current password.
3. The system prompts you for a new password. Enter a new password, or press the RETURN key to disable your current password.
4. The system prompts you to verify the password. Enter the new password to verify. (If the two entries of the new password do not match, the password does not change.)

The following guidelines are recommended to minimize the chances of passwords being discovered by trial-and-error or by exhaustive search:

- Make passwords at least six characters long.
- Avoid names or words that are readily associated with you.
- Change your passwords at least once every month.

To ensure that the above guidelines are met, use the /GENERATE[=value] qualifier. This qualifier generates random passwords of up to 12 characters in length. The system manager can require individual users to use the /GENERATE qualifier. For more information about this, see the description of the AUTHORIZE utility in the *VMS Authorize Utility Manual.*

QUALIFIERS

/GENERATE[=value]

Generates a list of 5 random passwords. Press RETURN to repeat the procedure until a suitable password appears.

Value is a number from 1 to 10 that restricts the length of the password. For any value *n*, SET PASSWORD generates passwords of from n to (n+2) characters long.

If no value is specified, SET PASSWORD uses a default value of 6, and generates passwords from 6 to 8 characters long. Values greater than 10 are not accepted and produce errors.

If your system manager has established a minimum password length for your account, SET PASSWORD/GENERATE=n compares that length with the length of the optional value, and uses the larger of the two values.

/SECONDARY

Creates or allows you to replace a secondary password. The procedure is the same as setting your primary password.

Once a secondary password has been established, you will receive two PASSWORD: prompts when logging in. The primary password should be typed in first, followed by the secondary password.

To remove your secondary password, press the RETURN key when SET PASSWORD/SECONDARY prompts you for a new password and verification. After you do this, you will receive a single PASSWORD: prompt when logging in.

Secondary passwords make it possible to set up an account that requires two different people to access it. Each person knows one of the two passwords, and both passwords are required to successfully log in.

The /SECONDARY and /SYSTEM qualifiers are incompatible.

/SYSTEM

Requires both SECURITY and CMKRNL privileges.

Changes the system password, rather than a user password.

If a terminal line has the system password (SYSPWD) characteristic set, no terminal prompts are sent to that terminal until the system password is entered.

A system password is valid only for the node it is set on. In a VAXcluster, each node can have a different system password.

The /SYSTEM and /SECONDARY qualifiers are incompatible. Refer to the *Guide to VMS System Security* for more information about the use of system passwords.

EXAMPLE

```
$ SET PASSWORD
Old password: HONCHO
New password: BIG_ENCHILADA
Verification: BIG_ENCHILADA
```

In response to the SET PASSWORD command, the system first prompts for the old password and then for the new password. The system then prompts again for the new password to verify it. The password changes if the user is authorized to change this account's password, if the old password is given correctly, and if the new password is given identically twice. Otherwise, an error message appears and the password remains unchanged.

In a real session, neither the old password nor the new password and its verification appear on the screen or paper.

SET PRINTER

Establishes the characteristics of a specific line printer. The defaults listed below are the defaults for an initially bootstrapped system.

Requires OPER privilege. If the printer is a spooled device, the logical I/O privilege (LOG_IO) is required to modify its characteristics.

FORMAT

SET PRINTER *printer-name[:]*

PARAMETER

printer-name[:]

Specifies the name of a line printer to set or modify its characteristics. If the printer has been set to /SPOOLED, the logical I/O privilege (LOG_IO) is required to modify its characteristics.

QUALIFIERS

/CR
/NOCR (default)

Controls whether the printer driver outputs a carriage return character. Use this qualifier for printers on which line feeds do not imply carriage returns.

Specify /NOCR for printers where the line feed, form feed, vertical feed, and carriage return characters empty the printer buffer. The /NOCR qualifier causes carriage return characters to be held back and output only if the next character is not a form feed or vertical tab. Carriage return characters are always output on devices that have the carriage return function characteristic set.

/FALLBACK
/NOFALLBACK (default)

Determines whether or not the printer attempts to translate characters belonging to the DEC Multinational Character Set into 7-bit equivalent representations. If a character cannot be translated, an underscore character is substituted.

If the /PASSALL qualifier is in effect, it has precedence over the /FALLBACK qualifier.

/FF (default)
/NOFF

Indicates whether the printer performs a mechanical form feed. Use the /NOFF qualifier when the printer does not automatically perform mechanical form feeds. This qualifier allows the driver to convert form feeds into multiple line feeds and to output them.

/LA11

Specifies the printer as an LA11. This qualifier provides information for the SHOW PRINTER command, which, in turn, provides the user with information about specific printers. If no printer type is specified, LP11 is assumed.

/LA180
Specifies the printer as an LA180. This qualifier provides information for the SHOW PRINTER command, which, in turn, provides the user with information about specific printers. If no printer type is specified, LP11 is assumed.

/LOG
/NOLOG (default)
Determines whether information confirming the printer setting is displayed at the terminal from which the SET PRINTER command was entered.

/LOWERCASE
/NOLOWERCASE
Indicates whether the printer prints both uppercase and lowercase letters or only uppercase. When the operator specifies the /NOLOWERCASE qualifier, all letters are translated to uppercase.

The /[NO]LOWERCASE and /[NO]UPPERCASE qualifiers are complementary; that is, /LOWERCASE is equivalent to /NOUPPERCASE, and /NOLOWERCASE is equivalent to /UPPERCASE.

/LP11 (default)
Specifies the printer as an LP11. This qualifier provides information for the SHOW PRINTER command, which, in turn, provides the user with information about specific printers. LP11 is the default printer type.

/PAGE=lines-per-page
Establishes the number of lines per page on the currently installed form; the number of lines can range from 1 to 255 and defaults to 64. The printer driver uses this value to determine the number of line feeds that must be entered to simulate a form feed.

/PASSALL
/NOPASSALL (default)
Controls whether the system interprets special characters or passes them as 8-bit binary data.

If you specify /PASSALL, the printer does not expand tab characters to spaces, fill carriage return or line feed characters, or recognize control characters.

/PRINTALL
/NOPRINTALL (default)
Controls whether the line printer driver outputs printable 8-bit multinational characters.

/TAB
/NOTAB (default)
Controls how the printer handles TAB characters. The /NOTAB qualifier expands all tab characters to spaces and assumes tab stops at eight character intervals.

Use the /TAB qualifier when you do not want the system to convert tabs to spaces, but want the printer to process the tab characters. The VMS operating system requires that printers expand tabs at eight-character intervals.

/TRUNCATE (default)
/NOTRUNCATE

Controls whether the printer truncates data exceeding the value specified by the /WIDTH qualifier. Note that the /TRUNCATE and /WRAP qualifiers are incompatible.

/UNKNOWN

Specifies the printer as nonstandard. This qualifier provides information for the SHOW PRINTER command, which, in turn, provides the user with information about specific printers. If no printer type qualifier is specified, LP11 is assumed.

/UPPERCASE
/NOUPPERCASE

Indicates whether the printer prints both uppercase and lowercase letters or only uppercase ones. When you specify /UPPERCASE, all letters are translated to uppercase.

The /[NO]UPPERCASE and /[NO]LOWERCASE qualifiers are complementary; that is, /UPPERCASE is equivalent to /NOLOWERCASE, and /NOUPPERCASE is equivalent to /LOWERCASE.

/WIDTH=n

Establishes the number of characters per output line on currently installed forms. The width, n, can range from 0 through 65535 for LP11 controllers, and from 0 through 255 for DMF32 controllers. The default value is 132 characters per line.

/WRAP
/NOWRAP (default)

Controls whether the printer generates a carriage return/line feed when it reaches the end of a line.

If the /NOWRAP qualifier is specified, the printer writes characters out in the last position on the line.

If the /WRAP qualifier is specified, the terminal generates a carriage return/line feed whenever the end of a line is reached.

Note that the /TRUNCATE and /WRAP qualifiers are incompatible.

EXAMPLES

1 `$ SET PRINTER/PAGE=60/WIDTH=80 LPA0:`

The SET PRINTER command in this example establishes the size of an output page as 60 lines and the width of a line as 80 characters for printer LPA0.

2 `$ SET PRINTER/LA11 LPB0:`

The SET PRINTER command in this example establishes the line printer LPB0 as an LA11 printer.

3 `$ SET PRINTER/LOWERCASE LPA0:`

The SET PRINTER command in this example requests that lowercase printing be enabled on line printer LPA0.

SET PROCESS

Changes the execution characteristics associated with the specified process for the current terminal session or job. If no process is specified, changes are made to the current process.

Requires GROUP privilege to change other processes in the same group. Requires WORLD privilege to change processes outside your group.

FORMAT

SET PROCESS *[process-name]*

PARAMETER

process-name

Requires that you own the process or that you have GROUP privilege and that the process is in your group.

Specifies the name of the process for which the characteristics are to be changed. The process name can contain from 1 to 15 alphanumeric characters. The default is the current process. Compatible only with the /PRIORITY, /RESUME, and /SUSPEND qualifiers.

You cannot specify the process-name for a process outside of your group. To change the characteristics of a process outside of your group, you must use the qualifier /IDENTIFICATION=pid. The process name parameter is ignored. If you include neither the process name nor the /IDENTIFICATION qualifier, the current process is assumed.

The process name parameter is limited to use only with the /PRIORITY, /RESUME, and /SUSPEND qualifiers.

QUALIFIERS

/DUMP
/NODUMP (default)

Causes the contents of the address space to be written to the file named SYS$LOGIN:IMAGEDUMP.DMP when an image terminates due to an unhandled error.

You can then analyze the dump with the ANALYZE/PROCESS_DUMP Utility.

/IDENTIFICATION=pid

Requires GROUP or WORLD privilege for processes other than your own.

Specifies the process identification value (PID) of the process for which characteristics are to be changed. Overrides the process-name parameter. Compatible only with the /PRIORITY, /RESUME, and /SUSPEND qualifiers.

The PID is assigned by the system when the process is created. When you specify a PID, you can omit the leading zeros.

If you use the /IDENTIFICATION qualifier, the process name parameter is ignored.

/NAME=string

Changes the name of the current process to a string of 1 through 15 characters.

/PRIORITY=n

Requires ALTPRI privilege to set the priority higher than your base priority.

Changes the priority for the specified process. If you do not have the ALTPRI privilege, the value you specify is compared to your current base priority, and the lower value is always used.

/PRIVILEGES=(privilege[,...])

Requires SETPRV privilege to enable a privilege you do not have.

Enables privileges for the process. For a list of process privileges, see Table A-1 in the *VMS DCL Concepts Manual*.

Use the SHOW PROCESS/PRIVILEGES command to determine what privileges are currently enabled.

/RESOURCE_WAIT
/NORESOURCE_WAIT

Enables resource wait mode so that the process waits for resources to become available. If you specify the /NORESOURCE_WAIT qualifier, the process receives an error status code when system dynamic memory is not available or when the process exceeds one of the following resource quotas: direct I/O limit, buffered I/O limit, or buffered I/O byte count (buffer space) quota.

/RESUME

Allows a process suspended by a previous SET PROCESS command to resume operation.

/SUSPEND[=SUPERVISOR]
/SUSPEND=KERNEL
/NOSUSPEND

Requires privileges as described in text.

Temporarily stops the process's activities. The process remains suspended until another process resumes or deletes it. The qualifiers /NOSUSPEND and /RESUME allow a suspended process to resume operation.

Specify either of the following keywords with /SUSPEND to produce different results:

Keyword	Result
SUPERVISOR (default)	Specifies that the named process is to be suspended to allow the delivery of Asynchronous System Traps (ASTs) at EXEC or KERNEL mode. Specifying this keyword is optional.
KERNEL	Specifies that the named process is to be suspended such that no asynchronous system traps (ASTs) can be delivered. To specify the KERNEL keyword, you must be in either kernel mode or exec mode, or have either CMKRNL or CMEXEC privilege enabled. Note that this was the default behavior of SET PROCESS/SUSPEND for Versions of VMS prior to Version 5.0.

Depending on the operation, the process from which you specify /SUSPEND requires privileges. You must have GROUP privilege to suspend another process in the same group, unless that process has the same UIC. You must have WORLD privilege to suspend any other process in the system.

Note that you can specify SET PROCESS/SUSPEND=KERNEL to override a previous SET PROCESS/SUSPEND=SUPERVISOR. SET PROCESS /SUSPEND=SUPERVISOR does not, however, override SET PROCESS /SUSPEND=KERNEL.

/SWAPPING (default)
/NOSWAPPING

Requires the user privilege PSWAPM to disable swapping for your process.

Permits the process to be swapped. By default, a process that is not currently executing can be removed from physical memory so that other processes can execute. If you specify /NOSWAPPING, the process is not swapped out of the balance set when it is in a wait state.

EXAMPLES

1
```
$ SET PROCESS/PRIVILEGE=EXQUOTA
```

The SET PROCESS command in this example assigns the current process the privilege of exceeding quotas.

2
```
$ SET PROCESS/NORESOURCE_WAIT
```

The SET PROCESS command in this example disables resource wait mode for the current process.

3
```
$ RUN/PROCESS_NAME=TESTER  CALC
%RUN-S-PROC_ID, identification of created process is 0005002F
$ SET PROCESS/PRIORITY=10  TESTER
```

The RUN command in this example creates a subprocess and gives it the name TESTER. Subsequently, the SET PROCESS/PRIORITY command assigns the subprocess a priority of 10.

4

```
$ SHOW PROCESS/SUBPROCESS

11-FEB-1988 15:17:28.41    VTA17:          User: REAGAN

Processes in this tree:

REAGAN *
  REAGAN_1
  REAGAN_2

$ SET PROCESS/SUSPEND REAGAN_1
$
```

The SET PROCESS/SUSPEND command in this example suspends the process REAGAN_1 such that ASTs can still be delivered to it. Because no keyword was specified, the /SUSPEND=SUPERVISOR version is assumed.

5

```
$ SHOW PROCESS/SUBPROCESS

17-FEB-1988 12:17:24.45    VTA13:          User: CHEESE

Processes in this tree:

CHEESE *
  CHEESE_1
  CHEESE_2
$ SET PROCESS/SUSPEND=KERNEL CHEESE_2
$
```

The SET PROCESS/SUSPEND=KERNEL command in this example suspends the process CHEESE_1 such that no ASTs can be delivered to it.

SET PROMPT

Replaces the default DCL prompt ($) with the specified string.

FORMAT

SET PROMPT*[=string]*

PARAMETER

string

Specifies the new prompt string. The following rules apply:

- All valid ASCII characters can be used.
- No more than 32 characters are allowed.
- To include spaces or lowercase letters, enclose the string in quotation marks. Otherwise, letters are automatically converted to uppercase; leading and trailing spaces are removed.

If you do not specify the string parameter with the SET PROMPT command, the default DCL prompt ($) is restored.

DESCRIPTION

The SET PROMPT command customizes prompts for your main process or a subprocess.

When a continued command is read from the terminal or an indirect command is read from a command procedure, an underscore is placed in front of the prompt string by DCL.

QUALIFIER

/CARRIAGE_CONTROL (default)
/NOCARRIAGE_CONTROL

Inserts carriage return and line feed characters before the prompt string. Type the qualifier after the string parameter.

EXAMPLE

```
$ SET PROMPT ="What's next?"
What's next? SHOW TIME
 31-DEC-1988 14:08:58
```

The SET PROMPT command in this example replaces the DCL prompt ($) with the phrase "What's next?". When you see the prompt on your screen, you can enter any DCL command. This example uses the SHOW TIME command.

SET PROTECTION

Establishes the protection that limits other users' access to a file or a group of files.

You cannot change the protection on a file on a network node other than the one you are currently logged in to.

FORMAT

SET PROTECTION*[=(code)] file-spec[,...]*

PARAMETERS

code

Defines the protection to be applied to the specified files. If you omit the code, the access is set to the current default protection.

The code is made up of the following components:

- Ownership category — SYSTEM, OWNER, GROUP, or WORLD. Each category can be abbreviated to its first character.
- Access category — R (READ), W (WRITE), E (EXECUTE), or D (DELETE). The access category is assigned to each ownership category. A null access specification means no access.

file-spec[,...]

Specifies one or more files for which the protection is to be changed. A file name and file type are required. If you omit a version number, the protection is changed only for the highest existing version of the file. Wildcard characters are allowed.

DESCRIPTION

All disk and tape volumes have protection codes that restrict access to the volume. The protection codes for disk and tape volumes are assigned with the INITIALIZE and MOUNT commands. They cannot be changed by the SET PROTECTION command.

For disk volumes, each file on the volume, including a directory file, can have a different protection associated with it. The SET PROTECTION command and other file manipulating commands let you define the protection for individual files.

Use the SET PROTECTION command to change or reset the access for one or more files. If you include a protection code, the file access is changed to that code. However, you cannot change the protection code for a file across a network.

If you omit both the protection code and the /PROTECTION qualifier, the file access changes to the default established by the SET PROTECTION /DEFAULT command. See the SET PROTECTION/DEFAULT command for information on how to change the default file access.

QUALIFIERS

/CONFIRM
/NOCONFIRM (default)

Controls whether the SET PROTECTION command displays the file specification of each file before applying the new protection, and requests you to confirm that the file's protection should be changed. To change the protection, type Y (YES) or T (TRUE) at the system prompt and press RETURN. If you enter anything else, such as N or NO, the file protection is not changed.

/LOG
/NOLOG (default)

Controls whether the system displays the file specification of each file for which the protection is changed as the command executes.

/PROTECTION=(code)

File-spec qualifier.

If you follow a file specification with the /PROTECTION qualifier, the code specified with /PROTECTION overrides the command's code parameter. The /PROTECTION qualifier lets you assign different protection codes to several files with a single SET PROTECTION command.

EXAMPLES

1

```
$ DELETE INCOME.DAT;3
%DELETE-W-FILNOTDEL, error deleting DISK1:[SMITH]INCOME.DAT;3
-RMS-E-PRV, insufficient privilege or file protection violation
$ SET PROTECTION=OWNER:D INCOME.DAT;3
$ DELETE INCOME.DAT;3
```

In this example, the file INCOME.DAT;3 has been protected against deletion. The SET PROTECTION command gives the owner the ability to delete the file INCOME.DAT;3.

2

```
$ SET PROTECTION -
$_PAYROLL.LIS/PROTECTION=(SYSTEM:R,OWNER:RWED,GROUP:RW),-
$_PAYROLL.OUT/PROTECTION=(SYSTEM:RWED,GROUP:RWED,W)
```

The SET PROTECTION command in this example changes the protection codes applied to two files. To the file PAYROLL.LIS, it gives the system READ access; the owner READ, WRITE, EXECUTE, and DELETE access; and users in the owner's group READ and WRITE access. To the file PAYROLL.OUT, it gives the system and group all types of access; the current access for the owner does not change, but the world category is denied all types of access.

```
3  $ SET PROTECTION A.DAT, B.DAT/PROTECTION=OWNER:RWED, C.DAT
```

The SET PROTECTION command in this example specifies that the file A.DAT receive the default protection established for your files. The existing protection for the file B.DAT is overridden, only for the owner category, to provide read, write, execute, and delete access. Note that no protection is specified for the file C.DAT at either the command or file level. Like A.DAT, C.DAT receives the default protection.

Since no version numbers are specified, the protection settings affect only the highest versions of the three files.

```
4  $ SET PROTECTION=OWNER:D -
   $_[MALCOLM.SUB1]SUB2.DIR/PROTECTION=GROUP:D
```

The SET PROTECTION command in this example changes the protection for the owner and group categories of the subdirectory [MALCOLM.SUB1.SUB2] to permit deletion. However, the protection for the world and system categories is not changed.

```
5  $ DIR/PROTECTION INCOME.DAT
   Directory DBA0:[SMITH]
   INCOME.DAT;2          (RWED,RWED,RWED,RWED)
   INCOME.DAT;1          (RWED,RWED,RWED,RWED)
   Total of 2 files.
   $ SET PROTECTION=(OWNER:RWE) INCOME.DAT;1
   $ PURGE
```

In this example, the file INCOME.DAT;1 has been protected against deletion by the owner. However, because the owner is also a member of the group and world categories, the file is still vulnerable to deletion. The subsequent PURGE command deletes INCOME.DAT;1.

To protect the file against deletion by you (the owner), you also need to protect the file against deletion by all outer access categories. The following command shows the proper way to do this.

```
$ SET PROTECTION=(OWNER:RWE,GROUP:RWE,WORLD:RWE) INCOME.DAT;1
```

SET PROTECTION/DEFAULT

Establishes the default protection to be applied to all files subsequently created.

FORMAT

SET PROTECTION*[=(code)]/DEFAULT*

PARAMETER

code

Defines the default protection to be applied to all files. To override this default protection use either the SET PROTECTION or CREATE commands. If you do not specify a protection code, the current default protection remains unchanged.

The code is made up of the following components:

- Ownership category — SYSTEM, OWNER, GROUP, or WORLD. Each category can be abbreviated to its first character.
- Access category — R (READ), W (WRITE), E (EXECUTE), or D (DELETE). The access category is assigned to each ownership category. A null access specification means no access.

EXAMPLE

```
$ SET PROTECTION=(GROUP:RWED,WORLD:R)/DEFAULT
```

The SET PROTECTION/DEFAULT command in this example sets the default protection to grant unlimited access to other users in the same group and read access to all users. The default protections for system and owner are not changed.

SET PROTECTION/DEVICE

Establishes the protection to be applied to a specific non-file-structured device. The protection for a device limits the type of access available to users. The /DEVICE qualifier is required.

Requires OPER privilege.

FORMAT

SET PROTECTION=*(ownership[:access],...)*/**DEVICE** *device-name[:]*

PARAMETERS

ownership

An ownership category — SYSTEM, OWNER, GROUP, or WORLD. Each category can be abbreviated to its first character. Any protection code category that the operator does not specify will remain unchanged.

access

An access category — R (READ), W (WRITE), L (LOGICAL I/O), and P (PHYSICAL I/O) —to be assigned to a specified type of owner. A null access specification means no access.

device-name[:]

Specifies the name of the non-file-structured device whose protection is to be set or modified.

DESCRIPTION

The same four user categories that apply to files also apply to devices: System, Owner, Group, and World.

- System—all users who have group numbers of 0 through 10 octal and users with physical or logical I/O privilege (generally, system managers, system programmers, and operators). The octal group numbers 0 through 10 for system users are the default group numbers. The group number parameter can be changed at system generation time to any octal value from 0 through 377.
- Owner—the user identification code (UIC) of the user who "owns" the device and is entering the SET PROTECTION command.
- Group—all users who have the same group number in their UICs as the owner of the device.
- World—all users who do not fall into the categories above.

For shareable devices (such as the LPA11-K), each user category can be allowed or denied one of the following types of access:

- READ—the right to issue read requests to the device
- WRITE—the right to issue write requests to the device
- LOGICAL I/O—the right to issue logical I/O requests to the device
- PHYSICAL I/O—the right to issue physical I/O requests to the device

For nonshareable devices, such as terminals and card readers, each category of user can either be allowed or denied access to allocate and assign channels to the device. The READ category controls whether a user can allocate and assign channels to the device. All other categories are not relevant for nonshareable devices.

Any combination of access types can be specified for any category of user. When the operator specifies a user access code, the code must be abbreviated to one character. The abbreviations are as follows:

Read	R
Write	W
Logical I/O	L
Physical I/O	P

The ownership and access categories can be specified in any order. If you specify an ownership category without including any access code, that category of user is denied all types of access. When you specify one or more access categories for a user category, that user category receives only those specified types of access. If you omit a user category, the access for that category is unchanged.

To specify a protection code, separate the user category from the access type with a colon. To specify more than one user category, separate each category by a comma and enclose the entire protection code specification in parentheses.

QUALIFIER

/OWNER_UIC=uic

Requests that the specified user identification code (UIC) be assigned ownership of the device for the purpose of access checks. The default owner is the UIC of the process entering the SET PROTECTION command.

EXAMPLES

1 `$ SET PROTECTION=(S:RWLP,O:RWLP,G,W)/DEVICE  LAA0:`

The command in this example requests that the protection for device LAA0 be set to allow all types of access to system processes and processes with the UIC of the current process. This command also denies access to anyone else.

2 `$ SET PROTECTION=(S,O:RWLP,G,W)/DEVICE/OWNER_UIC=[103,4] LAB0:`

The command in this example requests that the protection for device LAB0 be set to allow all types of access to processes with a UIC of [103,4]. This command also denies access to anyone else.

3 `$ SET PROTECTION=(S:R,O,G,W)/DEVICE/OWNER_UIC=[1,4] TTA1:`

The command in this example requests that the protection for the terminal TTA1 be set to allow only system processes to allocate the device. This command also denies access to anyone else. This type of protection is recommended for interactive terminals if system security is necessary. Note that the above protection code restricts which users can allocate the device, but the protection does not restrict users from logging in to the device.

SET QUEUE

Changes the current status or attributes of the specified queue.

Requires OPER privilege or EXECUTE (E) access to the specified queue.

FORMAT

SET QUEUE *queue-name[:]*

PARAMETER

queue-name[:]

Specifies the name of an execution queue or a generic queue.

DESCRIPTION

After a printer or batch queue has been created by the INITIALIZE/QUEUE command, using the SET QUEUE command changes the attributes or status of the queue without having to stop the queue.

The defaults for the SET QUEUE parameters depend on the parameters of the queue when it was initialized. For example, the default for /JOB_LIMIT with INITIALIZE/QUEUE is 1. However, if the queue you are altering was initialized with a job limit of 3, and if you did not specify the /JOB_LIMIT qualifier with the SET QUEUE command, the job limit would remain at 3 for that queue.

QUALIFIERS

/BASE_PRIORITY=n

Specifies the base process priority at which jobs are initiated from a batch queue. (You must stop and restart symbiont queues to change the symbiont priority for printer, terminal, or server queues.) The n specifier can be any decimal value from 0 through 15.

/BLOCK_LIMIT=([lowlim,]uplim)
/NOBLOCK_LIMIT (default)

Limits the size of print jobs that can be executed on a printer or terminal queue. This qualifier allows you to reserve certain printers for certain size jobs. You must specify at least one of the parameters.

The lower parameter specifies the minimum number of blocks accepted by the queue for a print job. The upper parameter specifies the maximum number of blocks accepted by the queue for a print job. If a job contains fewer blocks than the number specified by the lower parameter or more blocks than the number specified by the upper parameter, the job remains pending until the block limit for the queue is changed, enabling the job to execute.

If you specify only an upper limit for jobs, you can omit the parentheses. For example, /BLOCK_LIMIT=1000 means that only jobs with 1000 blocks or less execute in the queue. To specify only a lower job limit, use two consecutive quotation marks to indicate the upper specifier. For example, /BLOCK_LIMIT=(500,"") means any job with 500 or more blocks executes in the queue. You can specify both a lower and upper limit. For example, /BLOCK_LIMIT=(200,2000) means that jobs with less than 200 blocks or more than 2000 blocks do not run in the queue.

The /NOBLOCK_LIMIT qualifier cancels the /BLOCK_LIMIT setting previously established for that queue.

/CHARACTERISTICS=(characteristic[,...])
/NOCHARACTERISTICS

Specifies one or more characteristics for processing jobs on the queue. If only one characteristic is specified, you can omit the parentheses. Each time you specify /CHARACTERISTICS, all previously set characteristics are erased. Only the ones specified with the qualifier are now established for the queue.

Queue characteristics are installation-specific. The characteristic parameter can be either a value from 0 through 127 or a characteristic name that has been defined by the DEFINE/CHARACTERISTIC command.

When users include the /CHARACTERISTICS qualifier with a PRINT or SUBMIT command, all the characteristics they specify must also be specified for the queue executing the job. If not, the job remains pending in the queue until the queue characteristics are changed or until the users delete the entry with the DELETE/ENTRY command. Users need not specify every characteristic of a queue with a PRINT or SUBMIT command as long as the ones they specify are a subset of the characteristics set for that queue. The job also runs if no characteristics are specified.

The /NOCHARACTERISTICS qualifier cancels any /CHARACTERISTIC settings previously established for that queue.

/CLOSE

Prevents jobs from being entered in the queue through PRINT or SUBMIT commands or as a result of requeue operations. To allow jobs to be entered, use the /OPEN qualifier. Whether a queue accepts or rejects new job entries is independent of the queue's state (such as paused, stopped, stalled). When a queue is marked closed, jobs executing continue to execute and jobs already pending in the queue continue to be candidates for execution.

/CPUDEFAULT=time

Indicates the default CPU time limit for batch jobs. Time can be specified as delta time, 0, NONE (default), or INFINITE. You can specify up to 497 days of delta time. Both the value 0 and the keyword INFINITE allow unlimited CPU time (subject to the restrictions imposed by the /CPUMAXIMUM qualifier or the user authorization file). The keyword NONE specifies that no time limit is needed. The time cannot exceed the CPU time limit set by the /CPUMAXIMUM qualifier. See Section 1.4 of the *VMS DCL Concepts Manual* for information on specifying delta time.

/CPUMAXIMUM=time

Indicates the maximum CPU time limit for batch jobs. The /CPUMAXIMUM qualifier overrides the time limit specified in the user authorization file (UAF). Time can be specified as delta time, 0, NONE (default), or INFINITE. You can specify up to 497 days of delta time. Both the value 0 and the keyword INFINITE allow unlimited CPU time (subject to the restrictions imposed by the /CPUMAXIMUM qualifier or the user authorization file). The keyword NONE specifies that no time limit is needed. See Section 1.4 of the *VMS DCL Concepts Manual* for information on specifying delta times.

A CPU time limit for processes is specified by each user record in the system (UAF). You can also specify the following: a default CPU time limit for all jobs in a given queue and a maximum CPU time limit for all jobs in a given queue. Refer to Table DCL-1 for more information on specifying CPU time limits.

/DEFAULT=(option[,...])
/NODEFAULT

Establishes defaults for certain options of the PRINT command. Defaults are specified by the list of options. If you specify only one option, you can omit the parentheses. Once an option is set for the queue by the /DEFAULT qualifier, users do not have to specify that option in their PRINT commands. Possible options are as follows:

[NO]BURST[=keyword]	Specifies whether to print burst pages (flag pages printed over the paper's perforations for easy identification of individual files in a print job). The keyword ALL places burst pages before each printed file in the job. The keyword ONE places a burst page before the first printed file in the job.
[NO]FEED	Specifies whether a form feed is automatically inserted at the end of a page.
[NO]FLAG[=keyword]	Specifies whether to print flag pages. The keyword ALL places flag pages before each printed file in the job. The keyword ONE places a flag page before the first printed file in the job.
FORM=type	Specifies the default form for a printer, terminal, or server queue. If a job is not submitted with an explicit form definition, this form is used to process the job. The systemwide default form, form=0, is the default value for this keyword. See also /FORM_MOUNTED.
[NO]TRAILER[=keyword]	Specifies whether to print trailer pages. The keyword ALL places trailer pages after each printed file in the job. The keyword ONE places a trailer page after the last printed file in the job.

If you specify any of the keywords BURST, FLAG, TRAILER without specifying a value, the value ALL is used by default.

/DESCRIPTION=string
/NODESCRIPTION (default)

A string of up to 255 characters used to provide operator-supplied information about the queue.

If the string contains alphanumeric, underscore, or dollar sign characters it must be enclosed in quotation marks (").

The /NODESCRIPTION qualifier removes any descriptive text that may have been associated with the queue.

/DISABLE_SWAPPING
/NODISABLE_SWAPPING (default)

Controls whether batch jobs executed from a queue can be swapped in and out of memory.

/ENABLE_GENERIC (default)
/NOENABLE_GENERIC

Specifies whether files queued to a generic queue that does not have specific targets can be placed in this execution queue for processing.

/FORM_MOUNTED=type

Specifies the form type for a printer, terminal, or server queue. If the stock of the mounted form is not identical to the stock of the default form, as indicated by the DCL command qualifier /DEFAULT=FORM=type, all jobs submitted to this queue without an explicit form definition enter a pending state. If a job is submitted with an explicit form and the stock of the explicit form is not identical to the stock of the mounted form, then the job enters a pending state. In both cases, jobs remain pending until the stock of the mounted form of the queue is identical to the stock of the form associated with the job.

To specify the form type, use a numeric value or a form name that has been defined by the DEFINE/FORM command. Form types are specific to each installation.

/JOB_LIMIT=n

Indicates the number of batch jobs that can be executed concurrently from the queue.

/OPEN

Allows jobs to be entered in the queue through PRINT or SUBMIT commands or as the result of requeue operations. To prevent jobs from being entered, use the /CLOSE qualifier. Whether a queue accepts or rejects new job entries is independent of the queue's state (such as paused, stopped, stalled).

/OWNER_UIC=uic

Requires OPER privilege.

Enables you to change the user identification code UIC of the queue. Specify the UIC using standard UIC format as described in Section 8.1 of the *VMS DCL Concepts Manual*.

/PROTECTION=(ownership[:access],...)

Requires OPER privilege.

Specifies the protection of the queue. By default, the queue protection is (SYSTEM:E, OWNER:D, GROUP:R, WORLD:W). If you include only one protection code, you can omit the parentheses.

/RECORD_BLOCKING
/NORECORD_BLOCKING

Determines whether the symbiont can concatenate (or block together) output records for transmission to the output device. If you specify /NORECORD_BLOCKING, the symbiont is directed to send each formatted record in a separate I/O request to the output device. For the standard VMS print symbiont, record blocking can have a significant performance advantage over single-record mode.

/RETAIN[=option]
/NORETAIN

Retains jobs in the queue in a completed status after they have executed. Possible options are as follows:

ALL	Retains all jobs in the queue after execution (default)
ERROR	Retains in the queue only jobs that complete unsuccessfully

The /NORETAIN qualifier enables you to reset the queue to the default.

/SCHEDULE=[NO]SIZE

Specifies whether pending jobs in a printer or terminal queue are scheduled for printing based on the size of the job. When /SCHEDULE=SIZE (the default) is in effect, shorter jobs print before longer ones. With /SCHEDULE=NOSIZE, jobs are printed in the order they were submitted, regardless of size.

If you enter this command while there are pending jobs in any queue, its effect on future jobs is unpredictable.

/SEPARATE=(option[,...])
/NOSEPARATE

Specifies the job separation defaults for a printer or terminal queue. The job separation options are as follows:

[NO]BURST	Specifies whether a burst page prints at the beginning of every job. Specifying BURST also results in a flag page being printed.
[NO]FLAG	Specifies whether a flag page prints at the beginning of every job.
[NO]TRAILER	Specifies whether a trailer page prints at the end of every job.
[NO]RESET=(module[,...])	Specifies a job reset sequence for the queue. The specified modules from the device control library are used to reset the device each time a job reset occurs.

/WSDEFAULT=n

Defines a working set default for a batch job. The value set by this qualifier overrides the value defined in the user authorization file (UAF) of any user submitting a job to the queue.

Specify a positive integer in the range 1 through 65,535, 0, or the word NONE as the value for n. If 0 or NONE is specified for n, the working set default value defaults to the value specified either in the UAF or by the SUBMIT command (if specified). For more information, refer to Table DCL–2.

/WSEXTENT=n

Defines a working set extent for the batch job. The value set by this qualifier overrides the value defined in the user authorization file (UAF) of any user submitting a job to the queue.

Specify a positive integer in the range 1 through 65,535, 0, or the word NONE as the value for n. If 0 or NONE is specified for n, the working set extent value defaults to the value specified either in the UAF or by the SUBMIT command (if specified). For more information, refer to Table DCL–2.

/WSQUOTA=n

Defines the working set page size (working set quota) for a batch job. The value set by this qualifier overrides the value defined in the user authorization file (UAF) of any user submitting a job to the queue.

Specify a positive integer in the range 1 through 65,535, 0, or the word NONE as the value for n. If 0 or NONE is specified for n, the working set quota value defaults to the value specified either in the UAF or by the SUBMIT command (if specified). For more information refer to Table DCL–2.

A working set default size and a working set quota (maximum size) are included in each user record in the system user authorization file (UAF), and can be specified for both individual jobs and for all jobs in a given queue. The decision table (Table DCL–2) shows the action taken for different combinations of specifications that involve working set size and working set quota values.

EXAMPLES

1

```
$ INITIALIZE/QUEUE/DEFAULT=BURST/FORM_MOUNTED=LETTER/START SYS$PRINT
      .
      .
      .
$ STOP/QUEUE/NEXT SYS$PRINT
$ SET QUEUE /DEFAULT=BURST/FORM_MOUNTED=MEMO SYS$PRINT
```

In this example, the queue is initialized with the INITIALIZE/QUEUE command to have the following attributes: a burst page preceding each file in the job and the form type LETTER. Later the queue is stopped with the STOP/QUEUE/NEXT command so that the current job finishes processing before the queue stops. The SET QUEUE command changes the form type to MEMO.

2

```
$ SET QUEUE/DEFAULT=FORM=LN01_PORTRAIT LN01_PRINT
```

In this example, the SET QUEUE command changes the default form to LN01_PORTRAIT for the LN01_PRINT queue.

3

```
$ SET QUEUE/CLOSE SYS$BATCH
```

In this example, the batch queue SYS$BATCH is modified to prevent jobs from being entered in this queue.

SET QUEUE/ENTRY

Changes the current status or attributes of a job that is not currently executing in a queue. The /ENTRY qualifier is required.

As of VMS Version 5.0, the SET QUEUE/ENTRY command is superseded by the SET ENTRY command. Note that the SET ENTRY command has the same qualifiers as the SET QUEUE/ENTRY command; only the command parameters are different. DIGITAL recommends usage of the SET ENTRY command.

Requires OPER privilege or EXECUTE (E) access to the specified queue. If you have DELETE (D) access to the specified job, you can alter the attributes for that job. In addition, the queue name parameter is optional.

FORMAT

SET QUEUE/ENTRY=*entry-number queue-name[:]*

SET RESTART_VALUE

Sets a value for the symbol BATCH$RESTART, used for restarting portions of batch jobs. If the system encounters the command interactively, no action is taken. Use the SET RESTART_VALUE command in command procedures.

FORMAT

SET RESTART_VALUE=*string*

PARAMETER

string

Specifies a string of up to 255 characters specifying the label at which the batch job should begin executing again.

DESCRIPTION

Use the SET RESTART_VALUE command in command procedures. SET RESTART_VALUE specifies that a job interrupted by the system restarts in the middle after the interruption. The command relies on the values of two DCL symbols: $RESTART and BATCH$RESTART. $RESTART is set to false if a batch job is executing for the first time. If the batch job is rerunning, because the system failed or because the job was requeued, the value for $RESTART is set to true.

BATCH$RESTART has no definition if no SET RESTART_VALUE command has ever been executed by the batch job. Otherwise, the BATCH$RESTART definition has the value of the string parameter that was used with the most recently executed SET RESTART_VALUE command.

If a batch job has SET RESTART_VALUE commands in it, but you want the job to run in its entirety, enter the SET QUEUE/ENTRY/NOCHECKPOINT command to clear the BATCH$RESTART.

EXAMPLES

1
```
$ SET RESTART_VALUE=FIRST_PART
```

The SET RESTART_VALUE command in this example sets the value of BATCH$RESTART to FIRST_PART.

```
2  $ SET RESTART_VALUE = FIRSTPART
   $ IF $RESTART THEN GOTO 'BATCH$RESTART
      .
      .
      .
   $ FIRSTPART:
   $ RUN PART1
      .
      .
      .
   $ SET RESTART_VALUE = SECONDPART
   $ SECONDPART:
   $ RUN PART2
      .
      .
      .
```

In this example, the first command in the procedure sets the BATCH$RESTART value to FIRSTPART. The second command states that, if $RESTART is true, proceed to the value contained in BATCH$RESTART. ($RESTART will be true only if the job has been executed before, that is, the job is being rerun after a crash or after having been requeued.)

The $FIRSTPART: line marks a target for the GOTO statement. The following line contains the command to run PART1.EXE.

The second SET RESTART_VALUE command sets the BATCH$RESTART value to SECONDPART. The following line marks another target for the GOTO statement. The last line shown contains the command to run PART2.EXE.

When the job is first submitted, the $RESTART value is false, so the IF—GOTO command line is ignored. If the job is stopped during the run of PART1.EXE, the value of BATCH$RESTART is FIRSTPART. When the job is reinitiated, the value of $RESTART is true. Thus, the IF—GOTO command is processed and transfers to the $FIRSTPART: line in the procedure. PART1.EXE is rerun.

If the job is stopped during the run of PART2.EXE, the value of BATCH$RESTART is SECONDPART. When the job is reinitiated, the value of $RESTART is true. In this instance, the IF—GOTO command transfers to the $ SECONDPART: line in the procedure so that PART2.EXE can be run. PART1.EXE is not rerun.

SET RIGHTS_LIST

Allows users to modify the process or system rights list. You must specify either /DISABLE or /ENABLE with the SET RIGHTS_LIST command.

FORMAT

SET RIGHTS_LIST *id-name[,...]*

PARAMETER

id-name[,...]

Specifies identifiers to be added to or removed from the process or system rights list. The id-name parameter is a string of 1 to 31 alphanumeric characters, underscores, and dollar signs; each name must contain at least one nonnumeric character.

DESCRIPTION

The SET RIGHTS_LIST command modifies identifiers in your current process rights list, the rights list of another process on the system, or the system rights list. Use the following guidelines to determine which privileges are required for each case.

- Adding new identifiers or modifying existing identifiers in your process rights list that do not have the DYNAMIC attribute requires CMKRNL privilege.
- Modifying the rights list of other processes on the system requires CMKRNL privilege and either GROUP or WORLD privilege.
- Modifying the system rights list requires both CMKRNL and SYSNAM privileges.

This command can also be used to add attributes to existing identifiers.

QUALIFIERS

/ATTRIBUTES=(keyword[,...])

Specifies attributes to be associated with the identifiers. Attributes may be added to new or existing identifiers. The following are valid keywords:

[NO]DYNAMIC	Indicates whether or not unprivileged holders of the identifiers may add or remove them from the process rights list. The default is NODYNAMIC.
[NO]RESOURCE	Indicates whether or not holders of the identifiers may charge resources to them. The default is NORESOURCE.

/DISABLE

Removes the identifiers from the process or system rights list. You cannot use /DISABLE with the /ENABLE qualifier.

/ENABLE

Adds the identifiers to the process or system rights list. You cannot use /ENABLE with the /DISABLE qualifier.

/IDENTIFICATION=pid

Specifies the process identification value (PID) of the process whose rights list is to be modified. The PID is assigned by the system when the process is created. When you specify a PID, you can omit the leading zeros.

If you specify the /IDENTIFICATION qualifier, you cannot use the /PROCESS qualifier. By default, if neither the /IDENTIFICATION nor the /PROCESS qualifier is specified, the current process is assumed. You cannot use /IDENTIFICATION with the /SYSTEM qualifier.

/PROCESS[=process-name]

Specifies the name of the process whose rights list is to be modified. The process name can contain from 1 to 15 alphanumeric characters.

If you specify the /PROCESS qualifier, you cannot use the /IDENTIFICATION qualifier. By default, if neither the /PROCESS nor the /IDENTIFICATION qualifier is specified, the current process is assumed.

You cannot use /PROCESS with the /SYSTEM qualifier.

/SYSTEM

Specifies that the desired operation (addition or removal of an identifier) be performed on the system rights list. You cannot use /SYSTEM with /PROCESS or /IDENTIFICATION.

EXAMPLES

1

```
$ SET RIGHTS_LIST/ENABLE/ATTRIBUTES=RESOURCE MARKETING
```

The SET RIGHTS_LIST command in this example adds the MARKETING identifier to the process rights list of the current process. Specifying the RESOURCE attribute allows holders of the MARKETING identifier to charge resources to it.

2

```
$ SET RIGHTS_LIST/ENABLE/SYSTEM PHYSICS101
%SYSTEM-F-NOCMKRNL, operation requires CMKRNL privilege
$ SET PROCESS/PRIVILEGES=(CMKRNL,SYSNAM)
$ SET RIGHTS_LIST/ENABLE/SYSTEM PHYSICS101
```

The SET RIGHTS_LIST command in this example adds the PHYSICS101 identifier to the system rights list. You must have both the CMKRNL and SYSNAM privileges to modify the system rights list.

SET RMS_DEFAULT

Defines default values for the multiblock and multibuffer counts, network transfer sizes, prolog level, and extend quantity used by VMS RMS for file operations.

If you set the default for either the multiblock count or the multibuffer count at 0, VMS RMS tries to use the process default value or the system default value, in that order. If these are set at 0, VMS RMS uses a default value of 1. Defaults are set for sequential, relative or indexed sequential file organizations on a process-only basis, unless a systemwide basis is requested.

FORMAT

SET RMS_DEFAULT

PARAMETERS

None.

DESCRIPTION

Multiblocking and multibuffering of file operations can enhance the speed of input/output operations with VMS RMS. The defaults set with the SET RMS_DEFAULT command are applied for all file operations that do not specify explicit multiblock or multibuffer counts.

For more information on multiblock and multibuffer operations, see the *VMS System Services Reference Manual.*

For indexed-sequential files, SET RMS_DEFAULT defines default prolog level options.

For sequential files, SET RMS_DEFAULT defines default extensions. If a default extension is not specified in your program, the process or system default is used.

For network operations, SET RMS_DEFAULT defines network buffer sizes for transfer.

QUALIFIERS

/BLOCK_COUNT=count

Specifies a default multiblock count (0 through 127) for record I/O operations *only,* where count is the number of blocks to be allocated for each I/O buffer.

For more information on multiblock count, see the description of the RAB$B_MBC in the *VMS Record Management Services Manual.*

/BUFFER_COUNT=count

Specifies a default multibuffer count (0 through 127) for file operations, where count is the number of buffers to be allocated .

When you use the /BUFFER_COUNT qualifier, you can use the /DISK, /INDEXED, /MAGTAPE, /RELATIVE, /SEQUENTIAL, and /UNIT_RECORD qualifiers to specify the types of file for which the default is to be applied. If /BUFFER_COUNT is specified without any of these qualifiers,

/SEQUENTIAL is assumed. If file type is not specified, the default is applied to sequential files.

For more information on multibuffer count, see the description of the RAB$B_MBF in the *VMS Record Management Services Manual.*

/DISK

Applies the specified defaults to disk file operations. Values applied using the /SEQUENTIAL qualifier take precedence over values applied using the /DISK qualifier.

/EXTEND_QUANTITY=n

Specifies the number of blocks (n) to extend a sequential file where n can range from 0 to 65535. If you do not specify /EXTEND_QUANTITY, VMS RMS calculates its own extend value. The /EXTEND_QUANTITY qualifier value is used when the program does not explicitly specify an extent quantity.

/INDEXED

Applies the multibuffer default to indexed file operations.

/MAGTAPE

Applies the multibuffer default to magnetic tape operations. Values applied using the /SEQUENTIAL qualifier take precedence over values applied using the /MAGTAPE qualifier.

/NETWORK_BLOCK_COUNT=count

Specifies a default block count (0 through 127) for network access to remote files, where count represents the number of I/O buffers that VMS RMS allocates for transmitting and receiving data.

For remote file access, the buffer size is negotiated between between VMS RMS and the remote system's file access listener (FAL) with the smaller of the two sizes being selected.

Thus, the /NETWORK_BLOCK_COUNT value places an upper limit on the network buffer size that is used. It also places an upper limit on the largest record that may be transferred to or from a remote file. In other words, the largest record that can be transferred must be less than or equal to this value.

If you omit the value or specify a value of 0, VMS RMS uses the systemwide block count value. If this value is also 0, VMS RMS uses a size of one block.

/PROLOG=n

Specifies a default prolog level for indexed sequential files where acceptable values for n are 0, 2 or 3. If 0 (default) is specified, VMS RMS sets an appropriate prolog level.

/RELATIVE

Applies the multibuffer default to relative file operations.

/SEQUENTIAL (default)

Applies the multibuffer default to sequential file operations (overrides values applied using either the /DISK, /MAGNETIC TAPE or /UNIT RECORD qualifiers.)

The /SEQUENTIAL qualifier is the default if you do not specify either /RELATIVE or /INDEXED .

/SYSTEM

Requires CMKRNL privilege.

Applies specified defaults on a systemwide basis to all file operations.

/UNIT_RECORD

Applies the multibuffer default to file operations on unit record devices. Values applied using the /SEQUENTIAL qualifier take precedence over values applied using the /UNIT_RECORD qualifier.

EXAMPLES

1

```
$ SET RMS_DEFAULT/BLOCK_COUNT=24
$ SHOW RMS

          MULTI-  |                MULTIBUFFER COUNTS                | NETWORK
          BLOCK   | Indexed  Relative           Sequential           |  BLOCK
          COUNT   |                       Disk   Magtape  Unit Record |  COUNT
Process    24     |    0         0          0       0         0       |    0
System     16     |    0         0          0       0         0       |    8

           Prolog    Extend Quantity
Process      0              0
System       0              0
```

The SET RMS_DEFAULT command in this example sets the multiblock count for disk file I/O at 16 for user programs that do not explicitly set the multiblock count and applies only to the current process.

2

```
$ SET RMS_DEFAULT/BUFFER_COUNT=8/MAGTAPE
$ SHOW RMS_DEFAULT

          MULTI-  |                MULTIBUFFER COUNTS                | NETWORK
          BLOCK   | Indexed  Relative           Sequential           |  BLOCK
          COUNT   |                       Disk   Magtape  Unit Record |  COUNT
Process    24     |    0         0          0       8         0       |    0
System     16     |    0         0          0       0         0       |    8

           Prolog    Extend Quantity
Process      0              0
System       0              0
```

The SET RMS_DEFAULT command in this example defines the default multibuffer count for I/O magnetic tape operations at 8.

3

```
$ SET RMS_DEFAULT/BUFFER_COUNT=7/NETWORK_BLOCK_COUNT=16/SYSTEM
$ SHOW RMS_DEFAULT

          MULTI-  |                MULTIBUFFER COUNTS                | NETWORK
          BLOCK   | Indexed  Relative           Sequential           |  BLOCK
          COUNT   |                       Disk   Magtape  Unit Record |  COUNT
Process    24     |    0         0          0       8         0       |    0
System     16     |    0         0          7       7         7       |    16

           Prolog    Extend Quantity
Process      0              0
System       0              0
```

The SET RMS_DEFAULT command in this example defines the systemwide default multibuffer count at 7 for all sequential file operations on disk,

magnetic tape, and unit record devices. The command also sets the network block count at 16.

4

```
$ SET RMS_DEFAULT/EXTEND=50/INDEXED/BUFFER_COUNT=5
$ SHOW RMS_DEFAULT

          MULTI-  |                MULTIBUFFER COUNTS                | NETWORK
          BLOCK   | Indexed  Relative           Sequential           |  BLOCK
          COUNT   |                        Disk   Magtape  Unit Record |  COUNT
Process    24     |    5         0          0       8         0       |    0
System     16     |    0         0          7       7         7       |   16

           Prolog    Extend Quantity
Process      0             50
System       0              0
```

The SET RMS_DEFAULT command in this example sets the default multibuffer count for I/O operations on indexed files at 5. It also defines the default extend quantity for sequential I/O operations at 50 blocks. These defaults apply only to disk operations for user programs that do not explicitly set the multiblock count and are limited to the current process.

5

```
$ SET RMS_DEFAULT/PROLOG=2
$ SHOW RMS_DEFAULT

          MULTI-  |                MULTIBUFFER COUNTS                | NETWORK
          BLOCK   | Indexed  Relative           Sequential           |  BLOCK
          COUNT   |                        Disk   Magtape  Unit Record |  COUNT
Process    24     |    5         0          0       8         0       |    0
System     16     |    0         0          7       7         7       |   16

           Prolog    Extend Quantity
Process      2             50
System       0              0
```

For the current process, the SET RMS_DEFAULT command in this example specifies Prolog 2 as default for indexed files.

SET SYMBOL

Controls access to local and global symbols in command procedures.

FORMAT SET SYMBOL

DESCRIPTION

The SET SYMBOL command controls access to local and global symbols in command procedures by treating symbols as undefined. Because all global and local symbols defined in an outer procedure level are accessible to inner procedure levels, it is often necessary to mask these symbols without deleting them.

The symbol scoping context is different for local and global symbols. Local symbols are procedure level dependent. This means that specifying SET SYMBOL/SCOPE=NOLOCAL causes all symbols defined at an outer procedure level to be inaccessible to the current procedure level and any inner levels. For example, if SET SYMBOL/SCOPE=NOLOCAL was specified at procedure levels 2 and 4, procedure level 2 can access only level 2 local symbols. Level 3 can access levels 2 and 3 local symbols, and level 4 can access only level 4 local symbols.

Global symbols are procedure-level independent. The current global symbol scoping context is applied to all subsequent procedure levels. Specifying /SCOPE=NOGLOBAL causes all global symbols to become inaccessible for all subsequent commands until either /SCOPE=GLOBAL is specified or the procedure exits to a previous level at which global symbols were accessible. In addition, specifying /SCOPE=NOGLOBAL prevents you from creating any new global symbols until /SCOPE=GLOBAL is specified.

When you exit a procedure level to return to a previous procedure, the symbol scoping context from the previous level is restored for both local and global symbols.

To display the current symbol scoping state, use the lexical function F$ENVIRONMENT(symbol_scope).

QUALIFIER

/SCOPE=(keyword,...)

Controls access to local and global symbols. Lets you treat symbols as being undefined. Possible keywords are as follows:

NOLOCAL	Causes all local symbols defined in outer procedure levels to be treated as being undefined by the current procedure and all inner procedure levels.
LOCAL	Removes any symbol translation limit set by the current procedure level.
NOGLOBAL	Causes all global symbols to be inaccessible to the current procedure level and all inner procedure levels unless otherwise changed.
GLOBAL	Restores access to all global symbols.

EXAMPLES

1 `$ SET SYMBOL/SCOPE=NOLOCAL`

In this example, all local symbols defined in outer procedure levels are now undefined by the current procedure and all inner procedure levels.

2 `$ SET SYMBOL/SCOPE=NOGLOBAL`

In this example, all global symbols are now inaccessible to the current procedure level and all inner procedure levels unless otherwise changed.

SET TERMINAL

Sets the characteristics of a terminal. Entering a qualifier changes a characteristic; omitting a qualifier leaves the characteristic unchanged.

FORMAT

SET TERMINAL *[device-name[:]]*

PARAMETER

device-name

Specifies the device name of the terminal. The default is SYS$COMMAND if that device is a terminal. If the device is not a terminal, an error message is displayed.

DESCRIPTION

The SET TERMINAL command modifies specific terminal characteristics for a particular application or overrides system default characteristics. (These defaults are defined at each installation, based on the most common type of terminal in use.) The default characteristics for terminals are listed in Table DCL–14.

The terminal characteristics, local or remote, are determined automatically by the terminal driver for terminals that have the modem characteristic enabled. These characteristics are not affected by the SET TERMINAL command. For example, when you successfully dial in to a VAX processor, you establish your terminal as remote. When you hang up, the terminal characteristic is set back to local.

The set of terminals supported by the VMS operating system includes a set of VT100-family terminals that support special DIGITAL ANSI characteristics and escape sequences. For a description of these special characteristics and escape sequences, see the *VMS I/O User's Reference Manual: Part I.*

Table DCL–14 Default Characteristics for Terminals

Name	Unk.	For.	LA12	LA34 LA38 LA100 LQP02	LA36	LA120	LA210 LN03 LN01K	VT05	VT52 VT55	VT101	VT102	VT100 VT105	VT125	VT131 VT132	VT173	VT200	VT300	PRO
ADVANCED_VIDEO	no	•	no	no	no	no	no	no	no	no	yes	@	@	@	yes	yes	yes	yes
ALTYPEAHD	no	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
ANSI_CRT	no	•	no	no	no	no	no	no	no	yes	yes	yes	yes	yes	yes	yes	yes	yes
APPLICATION_KEYPAD /NUMERIC_KEYPAD	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
AUTOBAUD	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
BLOCK_MODE	no	•	no	no	no	no	no	no	no	no	no	no	no	yes	yes	no	no	no
BRDCSTMBX	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
BROADCAST	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
CRFILL	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
DEC_CRT	no	•	no	no	no	no	no	no	no	yes	yes	yes	yes	yes	no	yes	yes	yes
DEC_CRT2	no	•	no	no	no	no	no	no	no	no	no	no	no	no	no	yes	yes	no
DEC_CRT3	no	•	no	no	no	no	no	no	no	no	no	no	no	no	no	yes	yes	no
DIALUP	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
DISCONNECT	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
DMA	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
ECHO	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
EDIT_MODE	no	•	no	no	no	no	no	no	no	no	yes	@	@	@	yes	yes	yes	yes
EIGHT_BIT	•	•	no	no	no	no	yes	no	no	no	no	no	no	no	no	yes	yes	no
ESCAPE	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
FALLBACK	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	no	no	no
FRAME	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
FORM	•	•	yes	yes	no	yes	yes	no	no	no	no	no	no	no	no	no	no	no
FULLDUP/HALFDUP	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
HANGUP	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
HARDCOPY/SCOPE	•	•	hard	hard	hard	hard	hard	scope	scope	scope	scope	scope	scope	scope	scope	scope	scope	scope
HOSTSYNC	•	•	•	•	•	•	•	•	•	•	•	•	•	yes	yes	•	•	•
INSERT/OVERSTRIKE	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
LFFILL	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	•	0
LINE_EDITING	no	•	no	no	no	no	no	no	no	•	•	•	•	•	no	•	•	•
LOCAL_ECHO	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
LOWERCASE/UPPERCASE	•	•	low	low	low	low	low	up	low	low	low	low	low	low	low	low	low	low

• Indicates that the current setting is not affected by terminal type.
@ Optional terminal feature.

ZK-5170-86

Table DCL–14 (Cont.) Default Characteristics for Terminals

Name	Unk.	For.	LA12	LA34 LA38 LA100 LQP02	LA36	LA120	LA210 LN03 LN01K	VT05	VT52 VT55	VT101	VT102	VT100 VT105	VT125	VT131 VT132	VT173	VT200	VT300	PRO
MODEM	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	yes	•
PAGE	•	•	66	66	66	66	66	20	24	24	24	24	24	24	16	24	24	24
PARITY	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
PRINTER_PORT	no	•	no	no	no	no	no	no	no	no	yes	@	yes	no	no	@	yes	yes
READSYNC	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
REGIS	no	•	no	no	no	no	no	no	no	no	no	no	yes	no	no	@	@	@
REMOTE	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
SECURE_SERVER	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
SET_SPEED	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
SIXEL_GRAPHICS	no	•	no	no	no	no	yes	no	no	no	no	no	yes	no	no	@	@	no
SOFT_CHARACTERS	no	•	no	no	no	no	no	no	no	no	no	no	no	no	no	@	@	no
SPEED	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
SYSPASSWORD	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
TAB	•	•	yes	no	no	yes	no	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes
TTSYNC	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
TYPE_AHEAD	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
WIDTH	•	•	80	132	132	132	80	72	80	80	80	80	80	80	80	80	80	80
WRAP	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•

ZK-5171-86

QUALIFIERS

/ADVANCED_VIDEO
/NOADVANCED_VIDEO

Specifies that the terminal has advanced video attributes and is capable of 132-column video. If the terminal width is set to 132 columns and /ADVANCED_VIDEO is enabled, the terminal page limit is set to 24 lines. If /NOADVANCED_VIDEO is enabled, the terminal page limit is set to 12 lines.

/ALTYPEAHD

Sets the size of the type-ahead buffer when used with the /PERMANENT qualifier. You should specify SET TERMINAL/PERMANENT/ALTYPEAHD in the SYS$SYSTEM:STARTUP.COM for those communication lines that require this capability.

To use this feature interactively, specify SET TERMINAL/PERMANENT /ALTYPEAHD. This specification is effective at your next login.

/ANSI_CRT (default)
/NOANSI_CRT

Specifies whether the terminal conforms to ANSI CRT programming standards. Since ANSI standards are a proper subset of the DEC_CRT characteristics, the default for all VT100-family terminals is /ANSI_CRT.

/APPLICATION_KEYPAD

Specifies that the keypad is to be set to APPLICATION_KEYPAD mode, which allows you to enter DCL commands defined with the DEFINE/KEY command. By default, the terminal is set to NUMERIC_KEYPAD mode.

/AUTOBAUD
/NOAUTOBAUD

Specifies whether the terminal baud rate is set when you log in and sets the default terminal speed to 9600. You must press the RETURN key two or more times at intervals of at least one second for the baud rate to be correctly determined. If you press a key other than RETURN, /AUTOBAUD might detect the wrong baud rate. If this happens, wait for the login procedure to time out before continuing. The /AUTOBAUD qualifier must be used with the /PERMANENT qualifier.

The valid baud rates are as follows:

110	1200	4800
150	1800	9600
300	2400	19200
600	3600	

/BLOCK_MODE
/NOBLOCK_MODE

Performs block mode transmission, local editing, and field protection.

/BRDCSTMBX
/NOBRDCSTMBX

Sends broadcast messages to an associated mailbox if one exists.

/BROADCAST (default)
/NOBROADCAST

Enables reception of broadcast messages (such as those issued by MAIL and REPLY). Specify the /NOBROADCAST qualifier when you are using a terminal as a noninteractive device or when you do not want special output to be interrupted by messages. Use SET BROADCAST to exclude certain types of messages from being broadcast, rather than eliminating all messages.

/CRFILL[=fill-count]

Generates the specified number of null characters after each carriage return before transmitting the next meaningful character (to ensure that the terminal is ready for reception). The value must be an integer in the range 0 through 9. The default is /CRFILL=0.

/DEC_CRT[=(value1,value2,value3)]
/NODEC_CRT[=(value1,value2,value3)]

Specifies that the terminal conforms to DIGITAL VT100-, VT200-, or VT300-family standards and supports the minimum standards, including the additional DIGITAL escape sequences.

One of the following three optional values may be specified:

1 (default)	Requests that the DEC_CRT terminal characteristic be set.
2	Requests that the DEC_CRT2 terminal characteristic be set.
3	Requests that the DEC_CRT3 terminal characteristic be set. A level 3 terminal is described as follows: • Supports a status line (line 25, at the bottom of the screen) • Supports the IOS Latin-1 character set • Has terminal state interrogation (describes what state your terminal is in)

Note that DEC_CRT2 and DEC_CRT3 are supersets of DEC_CRT. Clearing DEC_CRT causes DEC_CRT2 and DEC_CRT3 to be cleared. Similarly, setting DEC_CRT3 will cause all subsets of DEC_CRT3 (including ANSI_CRT) to be set.

/DEVICE_TYPE=terminal-type

Informs the system of the terminal type and sets characteristics according to the device type specified. You can specify any of the following terminal types:

UNKNOWN	LA34
FT1 - FT8	LA38
LA12	LA100
LA36	LQP02
LA120	VT125
LN03	LN01K
VT05	VT131
VT52	VT132
VT55	VT173
VT100	VT200
VT101	PRO_SERIES
VT102	LA210
VT105	VT300

The default characteristics for the VT100, VT102, and VT125 series terminals are as follows:

/ADVANCEDVIDEO	/CRFILL=0	/LFFILL=0	/SPEED=9600
/NOALTYPEAHD[1]	/ECHO	/LOWERCASE	/TAB
/ANSI_CRT	/NOEIGHT_BIT	/NODMA	/TTSYNC
/NOAUTOBAUD	/NOESCAPE	/PAGE=24	/TYPE_AHEAD
/NOBLOCK_MODE	/NOFORM	/NOPARITY	/WIDTH=80
/NOBRDCSTMBX	/FULLDUP	/NOPASTHRU	/WRAP
/BROADCAST	/NOHOSTSYNC	/NOREADSYN	

[1]This is the default characteristic set by the system and is not a valid qualifier for your use.

The terminal types and characteristics that can be set are described in Table DCL–14.

/DIALUP
/NODIALUP (default)

Specifies that the terminal is a dial-up terminal.

/DISCONNECT
/NODISCONNECT (default)

Specifies that the process connected to this terminal not be disconnected if the line detects a hangup. The /DISCONNECT qualifier is valid only when /PERMANENT is specified.

/DISMISS
/NODISMISS (default)

Causes the terminal driver to ignore data causing a parity error (instead of terminating the currently outstanding I/O with an error status).

/DMA
/NODMA
Controls the use of direct memory access (DMA) mode on a controller that supports this feature.

/ECHO (default)
/NOECHO
Causes the terminal to display the input it receives. With /NOECHO, the terminal displays only system or user application output, or both.

/EDIT_MODE
/NOEDIT_MODE
Specifies that the terminal can perform ANSI-defined advanced editing functions.

/EIGHT_BIT
/NOEIGHT_BIT
Uses 8-bit ASCII protocol rather than 7-bit ASCII protocol. You can use the Terminal Fallback Facility (TFF) to set the 8-bit characteristic on terminals. If the terminal you specify has the TFF enabled, the /EIGHT_BIT qualifier has no effect. See the *VMS Terminal Fallback Utility Manual* for more information on terminal fallback.

/ESCAPE
/NOESCAPE (default)
Validates escape sequences.

/FALLBACK
/NOFALLBACK
Displays the 8-bit DIGITAL Multinational Character Set characters on the terminal in their 7-bit representation. The default depends on the /EIGHTBIT setting of the terminal. If the VMS Terminal Fallback Facility (TFF) is enabled, it activates the default character conversion tables for the named terminal. See the *VMS Terminal Fallback Utility Manual* for more information. If TFF is not enabled on your system, /FALLBACK has no effect and no error message is displayed.

/FORM
/NOFORM
Transmits a form feed rather than translating it into multiple line feeds.

/FRAME=n
Specifies the number of data bits that the terminal driver expects for every character that is input or output. The value of *n* can be from 5 through 8. The default value depends on the /PARITY and /EIGHTBIT settings of the terminal.

/FULLDUP (default)
/NOFULLDUP
Operates in full duplex mode. The /FULLDUP qualifier is equivalent to /NOHALFDUP.

/HALFDUP
/NOHALFDUP (default)

Operates in half duplex mode. The /HALFDUP qualifier is equivalent to /NOFULLDUP.

/HANGUP
/NOHANGUP (default)

May require LOG_IO or PHY_IO privilege depending on system generation parameter settings.

Controls whether the terminal modem is hung up when you log out.

/HARDCOPY
/NOHARDCOPY

Establishes the device as a hardcopy terminal and outputs a backslash (\) when the DELETE key is pressed. The /HARDCOPY qualifier is equivalent to /NOSCOPE.

/HOSTSYNC
/NOHOSTSYNC (default)

When you specify the /HOSTSYNC qualifier, the system stops transmission to the terminal (by generating a CTRL/S) when the input buffer is full and resumes transmission (by generating a CTRL/Q) when the input buffer is empty.

/INQUIRE

Sets the device type according to a response elicited from the terminal; the default is UNKNOWN. Works only on DIGITAL terminals, and not on the LA36 or VT05 terminals. Some VT100-family terminals, including the VT101 and VT105, return a VT100-type response. LA38 terminals respond as LA43 terminals.

You can include the SET TERMINAL/INQUIRE command in your LOGIN.COM file to automatically detect the terminal type.

CAUTION: This qualifier clears the type-ahead buffer. If the response sequence is unrecognized, no action message or error message is displayed. The /INQUIRE qualifier should be used only on DIGITAL terminals. However, the LA36 and VT05 terminals do not support this feature.

/INSERT

Sets the terminal to /INSERT mode. This feature allows you to insert characters when editing command lines. The default mode is /OVERSTRIKE, which allows you to type over the current character when editing a command line. Use CTRL/A to switch from one mode to the other.

/LFFILL[=fill-count]

Transmits to the terminal the specified number of null characters after each line feed before transmitting the next meaningful character (to ensure that the terminal is ready for reception). The value must be an integer in the range 0 through 9. The default is installation-dependent. See Table DCL–14 for a list of default terminal characteristics.

/LINE_EDITING
/NOLINE_EDITING

Enables advanced line-editing features for editing command lines: both RETURN and CTRL/Z are recognized as line terminators, as are escape sequences.

/LOCAL_ECHO
/NOLOCAL_ECHO (default)

Echoes characters locally (rather than the host echoing them) for command level terminal functions. (Do not use /LOCAL_ECHO with utilities that require control over echoing, such as line editing or EDT's screen mode.)

CAUTION: When logging in to terminals with /LOCAL_ECHO set, the VMS operating system has no control over the echoing of passwords.

/LOWERCASE
/NOLOWERCASE

Passes lowercase characters to the terminal. The /NOLOWERCASE qualifier translates all input to uppercase. /LOWERCASE is equivalent to /NOUPPERCASE.

/MANUAL

Indicates manual switching of terminal lines to dynamic asynchronous DDCMP lines when your local terminal emulator does not support automatic switching. The /MANUAL qualifier should be specified with the /PROTOCOL=DDCMP and /SWITCH=DECNET qualifiers.

/MODEM
/NOMODEM

Indicates that the terminal is connected to a modem or a cable that supplies standard EIA modem control signals. If your terminal has the MODEM characteristic, typing SET TERMINAL/NOMODEM automatically logs you out.

/NUMERIC_KEYPAD (default)

Specifies that the keypad is to be set to /NUMERIC_KEYPAD mode, which allows you to use the keys on the numeric keypad to type numbers and punctuation marks. In order to use the DEFINE/KEY facility, which allows you to enter DCL commands defined with the DEFINE/KEY command, set the terminal to /APPLICATION_KEYPAD. Specifies whether the keys of the numeric keypad are used to type numbers and punctuation marks (/NUMERIC_KEYPAD) or to enter DCL commands defined with the DEFINE /KEY command (/APPLICATION_KEYPAD).

/OVERSTRIKE (default)

Sets the terminal to /OVERSTRIKE mode. This feature allows you to type over the current character when you are editing a command line. Set your terminal to /INSERT if you want to insert characters when editing command lines. Use CTRL/A to switch from one mode to the other.

/PAGE[=lines-per-page]

For hardcopy terminals, specifies the number of print lines between perforations. (When the terminal reads a form feed, it advances the paper to the next perforation.) The value of n can be from 0 through 255 and defaults to 0 (which treats a form feed as a line feed).

/PARITY[=option]
/NOPARITY (default)
Passes data with odd or even parity, where option equals ODD or EVEN. If you specify /PARITY without an option, the value defaults to EVEN.

/PASTHRU
/NOPASTHRU (default)
Passes all data (including tabs, carriage returns, line feeds, and control characters) to an application program as binary data. The setting of /TTSYNC is allowed.

/PERMANENT
Requires LOG_IO or PHY_IO privilege.

Sets characteristics on a permanent basis, that is, over terminal sessions. However, the characteristics revert to their initial values if the system is halted and restarted. Use in a system start-up file to establish characteristics for all terminals on the system.

/PRINTER_PORT
/NOPRINTER_PORT
Specifies that the terminal has a printer port (an attribute not set by the SET TERMINAL/INQUIRE command). The default is installation-dependent. See Table DCL–14 for a list of default terminal characteristics.

/PROTOCOL=DDCMP
/PROTOCOL=NONE (default)
Controls whether the terminal port specified is changed into an asynchronous DDCMP line. The /PROTOCOL=NONE qualifier changes an asynchronous DDCMP line back into a terminal line. Note that /PROTOCOL=DDCMP is a permanent characteristic; therefore, the /PERMANENT qualifier is not required.

/READSYNC
/NOREADSYNC (default)
Uses the CTRL/S and CTRL/Q functions to synchronize data transmitted from the terminal.

The default is /NOREADSYNC; the system does not use CTRL/S and CTRL/Q to control reads to the terminal. The /READSYNC qualifier is useful for certain classes of terminals that demand synchronization or for special-purpose terminal lines where data synchronization is appropriate.

/REGIS
/NOREGIS
Specifies that the terminal understands REGIS graphic commands.

/SCOPE
/NOSCOPE
Establishes the device as a video terminal. /SCOPE is equivalent to /NOHARDCOPY.

/SECURE_SERVER
/NOSECURE_SERVER (default)

Causes the BREAK key on the terminal to log out the current process (except on a virtual terminal). With /SECURE_SERVER in effect, pressing the BREAK key when there is no current process initiates the login sequence. With /NOSECURE_SERVER in effect, the break is ignored.

On terminals set with /AUTOBAUD, with the /SECURE_SERVER qualifier in effect, pressing the BREAK key disconnects the current process but is not required to start a new login sequence. However, when /NOAUTOBAUD is set, the /SECURE_SERVER characteristic requires a break to initiate a new login sequence.

/SET_SPEED
/NOSET_SPEED

Requires either LOG_IO or PHY_IO privilege.

Allows the /SPEED qualifier to be used to change the terminal speed.

/SIXEL_GRAPHICS
/NOSIXEL_GRAPHICS

Specifies that the terminal is capable of displaying graphics using the sixel graphics protocol. The default is device-dependent. See Table DCL–14 for a list of default terminal characteristics.

/SOFT_CHARACTERS
/NOSOFT_CHARACTERS

Specifies that the terminal is capable of loading a user-defined character set. The default is device-dependent. See Table DCL–14 for a list of default terminal characteristics.

/SPEED=(input-rate,output-rate)

Sets the baud rate at which the terminal receives and transmits data. If the input and output rates are the same, specify /SPEED=rate.

Not all terminals support different input and output baud rates. For specific information on baud rates for your terminal, consult the manual for that terminal.

The default transmission rates are installation-dependent.

The valid values for input and output baud rates are as follows:

50	150	1800	4800
75	300	2000	7200
110	600	2400	9600
134	1200	3600	19200

/SWITCH=DECNET

Causes the terminal lines at each node to be switched to dynamic asynchronous DDCMP lines, when specified with the /PROTOCOL=DDCMP qualifier. Note that /SWITCH=DECNET is a permanent characteristic; therefore, the /PERMANENT qualifier is not required.

/SYSPASSWORD
/NOSYSPASSWORD (default)

Requires LOG_IO privilege.

Determines whether the terminal requires that a system password be entered before the user name prompt.

/TAB
/NOTAB

Does not convert tab characters to multiple blanks. The /NOTAB qualifier expands all tab characters to blanks and assumes tab stops at 8-character intervals. The default is device-dependent. See Table DCL–14 for a list of default terminal characteristics.

/TTSYNC (default)
/NOTTSYNC

Stops transmitting to the terminal when CTRL/S is pressed and resumes transmission when CTRL/Q is pressed.

/TYPE_AHEAD (default)
/NOTYPE_AHEAD

Accepts unsolicited input for the terminal to the limit of the type-ahead buffer.

When you specify /NOTYPE_AHEAD, the terminal is dedicated, and accepts input only when a program or the system issues a read to the terminal. Logins are disabled on a terminal with /NOTYPE_AHEAD set. When you specify /TYPE_AHEAD, the amount of data that can be accepted is governed by the size of the type-ahead buffer. That size is determined by system generation parameters.

/UNKNOWN

Specifies a terminal type that is unknown to the system, which then uses the default terminal characteristics for unknown terminals. For a summary of the settings, see Table DCL–14.

/UPPERCASE
/NOUPPERCASE

Translates lowercase to uppercase characters. The /UPPERCASE qualifier is equivalent to /NOLOWERCASE.

/WIDTH=characters-per-line

Specifies the maximum characters per line. This value must be an integer in the range 1 through 511. With /WRAP, the terminal generates a carriage return and line feed when the width specification is reached.

If the specified width on an ANSI terminal is 132, the screen is set to 132-character mode. If the terminal does not have advanced video option (AVO), the page length limit is set to 12 lines.

/WRAP (default)
/NOWRAP

Generates a carriage return and line feed when the value of /WIDTH is reached.

EXAMPLES

1
```
$ SET TERMINAL/DEVICE=VT102
```

In this example, the SET TERMINAL command establishes the current terminal as a VT102 terminal and sets the default characteristics for that terminal type.

2
```
$ SET TERMINAL/WIDTH=132/PAGE=60/NOBROADCAST
$ TYPE MEMO.DOC
    .
    .
    .
$ SET TERMINAL/DEVICE=LA36
```

In this example, the first SET TERMINAL command indicates that the width of terminal lines is 132 characters and that the size of each page is 60 lines. The /NOBROADCAST qualifier disables the reception of broadcast messages while the terminal is printing the file MEMO.DOC. The next SET TERMINAL command restores the terminal to its default state.

SET TIME

Resets the system clock, which is used both as a timer to record intervals between various internal events, and as a source clock for displaying the time of day.

Requires both OPER and LOG_IO privileges.

FORMAT

SET TIME*[=time]*

PARAMETER

time

Specifies a date in the format day-month-year, or a time in the format hour:minute:second.hundredth, or both. Day must be an integer in the range 1 through 31. Month must be JAN, FEB, MAR, APR, MAY, JUN, JUL, AUG, SEP, OCT, NOV, or DEC. Year must be an integer in the range 1858 through 9999. Hour must be an integer in the range 0 through 23. Minute must be an integer in the range 0 through 59. Second must be an integer in the range 0 through 59. Hundredth (of a second) must be an integer in the range 0 through 99. The hyphens, colons, and period are required delimiters. Delimit the date and time, when both are specified, with a colon. The syntax is sometimes specified as follows:

[dd-mmm-yyyy[:]] [hh:mm:ss.cc]

If the explicit time value is not specified, the interval system clock is automatically reset according to the time-of-year clock.

Note that the time-of-year clock is optional for some processors. For further information about the time-of-year clock, see the *VAX Hardware Handbook*.

EXAMPLES

1

```
$ SET TIME=31-DEC-1988:19:31:0.0
```

The SET TIME command in this example sets the date/time at December 31, 1988, 7:31 P.M.

2

```
$ SET TIME
$ SHOW TIME
31-DEC-1988 03:21:27.53
```

The SET TIME command in this example sets the system time according to the time-of-year clock. The SHOW TIME command requests a display of the current time.

SET UIC

Changes the user identification code (UIC) of your process. Use the SET UIC command to gain access to a restricted file, that is, a file contained in a directory whose protection restricts access to the owner of that directory.

Requires CMKRNL privilege.

FORMAT

SET UIC *uic*

PARAMETER

uic

Specifies a valid UIC. Brackets are required around the UIC. Specifies the group number and member number. Specify the UIC using standard UIC format as described in the section on UIC protection in the *VMS DCL Concepts Manual*.

EXAMPLES

1
```
$ SET UIC [370,10]
```

The SET UIC command in this example establishes your UIC as [370,10]. You can now read or modify any files whose access is restricted to this UIC.

2
```
$ SET UIC [214,4]
$ SET DEFAULT [ANDERSON]
```

The SET UIC command in this example sets your UIC to [214,4]; the SET DEFAULT command sets the default directory name to [ANDERSON].

3
```
$ SET UIC [GEORGE]
```

This example sets the UIC to be that of the user named GEORGE who is a member of the same group as the person entering the SET UIC command. Note the similarity of this UIC format to the directory name format. Be sure not to use a UIC where a directory specification is needed.

4
```
$ SET UIC [VMS,GEORGE]
```

This example sets the UIC to be that of the user named GEORGE who is a member of the VMS group. The person entering the SET UIC command need not be a member of the VMS group.

SET VERIFY

Controls whether command lines and data lines in command procedures are displayed at the terminal or printed in a batch job log. The information displayed by the SET VERIFY command can help you in debugging command procedures.

FORMAT

SET [NO]VERIFY*[=([NO]PROCEDURE, [NO]IMAGE)]*

PARAMETER

([NO]PROCEDURE, [NO]IMAGE)

Specifies one or both types of verification. Procedure verification causes each DCL command line in a command procedure to be written to the output device. Image verification causes data lines (input data that is included as part of the SYS$INPUT input stream) to be written to the output device.

By default, both types of verification are set or cleared with SET VERIFY and SET NOVERIFY. If you specify only one keyword, the other is not affected. If you specify only one keyword, omit the parentheses.

DESCRIPTION

By default, the SET VERIFY and SET NOVERIFY commands set or clear both types of verification. The default setting for command procedures executed interactively is SET NOVERIFY. System responses and error messages are, however, always displayed. The default for batch jobs is SET VERIFY.

If you use the SET VERIFY command to override the default setting, the system displays each command and data line in the command procedure as it reads it. When verification is in effect, the command interpreter displays each command line after it has completed initial scanning and before the command is parsed and executed. You see the results of symbol substitution performed during scanning, but not the results of symbol substitution performed during parsing and evaluation.

When you change the verification setting, the new setting remains in effect for all command procedures that you subsequently execute.

EXAMPLES

1

```
$ SET VERIFY
$ INDEX == "$INDEX.EXE
$ CONTENTS == "$CONTENTS.EXE
$ TABLE == "$TABLE.EXE
$ SET NOVERIFY
$ EXIT
```

Procedure and image verification are turned on at the beginning of the command procedure so that the system displays all the command and data lines in the procedure as it reads them. At the end of the procedure, the SET NOVERIFY command restores the system default (no procedure or image verification).

2

```
$ PROC_VER = F$ENVIRONMENT("VERIFY_PROCEDURE")
$ IMAGE_VER = F$ENVIRONMENT("VERIFY_IMAGE")
$ SET NOVERIFY
    .
    .
    .
$ TEMP = F$VERIFY(PROC_VER, IMAGE_VER)
```

This command procedure uses the lexical function F$ENVIRONMENT to save the current procedure and image verification setting. Then the SET NOVERIFY command turns off both procedure and image verification. Subsequently, the F$VERIFY function is used to restore the original verification settings.

3

```
$ SET VERIFY
$ @TEST
$ RUN AVERAGE
1
2
3
$ EXIT
```

In this example, the SET VERIFY command turns procedure and image verification on. When the command procedure TEST.COM is executed interactively, the command lines and the data lines for the program AVERAGE are displayed on the terminal. The data lines were entered in the command procedure on lines that did not begin with the DCL prompt.

4

```
$ SET VERIFY = PROCEDURE
```

In this example, procedure verification is turned on. If image verification was on, it remains on; if image verification was off, it remains off.

5

```
$ SET VERIFY
$ COUNT = 1 $ IF P'COUNT' .NES. "" THEN GOTO &P'COUNT'
    .
    .
    .
$ EXIT
```

When this command procedure is executed interactively, the SET VERIFY command causes the command and data lines to be displayed. Symbols that are substituted during the first phase of symbol substitution (such as 'COUNT') are displayed by the SET VERIFY command, but other symbols are not. The following lines are displayed when this procedure is executed interactively:

```
$ COUNT = 1 $ IF P1 .NES. "" THEN GOTO &P1
    .
    .
    .
```

Although these values are not displayed, the value for P1 is substituted during the third phase of symbol substitution, and the value for &P1 is substituted during the second phase.

SET VOLUME

Changes the characteristics of one or more mounted Files–11 volumes.

Requires WRITE (W) access to the index file on the volume. If you are not the owner of the volume, requires either a system UIC or SYSPRV privilege.

FORMAT

SET VOLUME *device-spec[:][,...]*

PARAMETER

device-name[:][,...]

Specifies the name of one or more mounted Files–11 volumes.

QUALIFIERS

/ACCESSED[=n]

Requires OPER privilege.

Specifies the number of directories to be maintained in system space for ready access. You can specify a number (n) in the range of 0 through 255. If you specify the qualifier /ACCESSED and omit the number of directories, a default value of 3 is used. If you specify a value greater than the current value, the new value is effective immediately; otherwise, the new value is not effective until the next time the volume is mounted.

/DATA_CHECK[=(option[,...])]

Defines a default for data check operations following all reads and writes to the specified volume. (If you do not specify the /DATA_CHECK qualifier, no checks are made.) Possible keywords are as follows:

READ	Performs checks following all read operations
WRITE	Performs checks following all write operations (default)

/ERASE_ON_DELETE
/NOERASE_ON_DELETE (default)

Determines whether the space occupied by a file is overwritten with a system specified pattern when a file on the volume is deleted.

/EXTENSION[=n]

Specifies the number of blocks to be used as a default extension size for all files on the volume. You can specify a number (n) in the range of 0 through 65,535. If you specify the /EXTENSION qualifier without specifying a value, a default value of 0 (the VMS RMS default) is used.

For example, during an update operation, the extension default is used when a file increases to a size greater than its initial default allocation.

/FILE_PROTECTION=(code)

Sets the default protection to be applied to all files on the specified disk volume. Specify ownership as SYSTEM, OWNER, GROUP, or WORLD and access as R (READ), W (Write), E (EXECUTE), or D (DELETE). A null access specification means no access.

Note that this attribute is not used while the volume is in use on a VMS operating system, but the attribute is provided to control the process use of the volume on RSX-11M systems. The VMS operating system always uses the default file protection; the protection can be changed with the DCL command SET PROTECTION/DEFAULT.

/HIGHWATER_MARKING
/NOHIGHWATER_MARKING

Determines whether the File Highwater Mark (FHM) volume attribute is set. The FHM attribute guarantees that a user cannot read data that was not written by the user. Applies to Structure Level 2 volumes only.

/LABEL=volume-label

Specifies a 1 through 12-character alphanumeric name to be encoded on the volume. Characters are automatically changed to uppercase. The specified label remains in effect until it is explicitly changed; dismounting the volume does not affect the label.

/LOG
/NOLOG (default)

Determines whether the volume specification of each volume is displayed after the modification.

/MOUNT_VERIFICATION
/NOMOUNT_VERIFICATION

Determines whether mount verification is enabled. Mount verification prevents interruption to user input/output operations and notifies the operator of problems with the disk.

/OWNER_UIC[=uic]

Sets the owner UIC of the volume to the specified UIC. The default UIC is that of the current process. Brackets are required around the UIC. Specify the UIC using standard UIC format as described in Section 8.1 of the *VMS DCL Concepts Manual*.

/PROTECTION=(code)

Specifies the protection to be applied to the volume. The ownership categories are SYSTEM, OWNER, GROUP, and WORLD; the access categories are R (READ), W (WRITE), E (EXECUTE), and D (DELETE). The default protection is all types of access by all categories of user.

When you specify a protection code, access type E (EXECUTE) indicates create access.

/REBUILD

Recovers caching limits for a volume that was improperly dismounted. If a disk volume was dismounted improperly (such as during a system failure), and was then remounted with the MOUNT/NOREBUILD command, you can use SET VOLUME/REBUILD to recover the caching that was in effect at the time of the dismount.

/RETENTION=(min[,max])

Specifies the minimum and maximum retention times to be used by the file system to determine the expiration date for files on the volume. When a file is created, its expiration date is set to the current time + maximum. Each time

the file is accessed, the current time is added to the minimum time. If the sum is greater than the expiration date, a new expiration date is computed.

If you omit the max value, a default value that is the smaller of (2 x min) or (min + 7) days is used. For example, /RETENTION=3- is the same as /RETENTION=(3-,6-), while /RETENTION=10- is the same as /RETENTION=(10-,17-).

/UNLOAD (default)
/NOUNLOAD

Specifies whether the volume is unloaded (spun down) when the DCL command DISMOUNT is entered.

/USER_NAME[=user-name]

Specifies a user name of up to 12 alphanumeric characters to be recorded on the volume. The default name is the current process user name.

/WINDOWS[=n]

Specifies the number of mapping pointers to be allocated for file windows. The value of *n* can be from 7 through 80; the default value is 7.

EXAMPLES

1
```
$ SET VOLUME/DATA_CHECK=(READ,WRITE) DBC5
```

The SET VOLUME command in this example requests that data checks be performed following all read and write operations to DBC5.

2
```
$ SET VOLUME/FILE_PROTECTION=(S:RWED,O:RWED,G:RE,W:RE) DBC5
```

The SET VOLUME command in this example sets the default protection to be applied to all files created on volume DBC5. System and owner are granted all types of access; group and world are permitted only to read and execute files on DBC5.

3
```
$ SET VOLUME/LABEL=LICENSES DBC5
```

The SET VOLUME command in this example encodes the label LICENSES on the volume DBC5. Note that if characters in labels are entered in lowercase, they are changed to uppercase by the /LABEL qualifier.

4
```
$ SET VOLUME/ACCESSED=25/USER_NAME=MANAGER/LOG DBA0:
```

The SET VOLUME command in this example specifies that 25 directories are to be maintained in system space for ready access for the volume DBA0. The command also assigns the user name MANAGER to the volume and displays the volume specification after the volume is modified.

5
```
$ SET VOLUME/REBUILD/LOG NODE$DBA2:
%SET-I-MODIFIED, _NODE$DBA2: modified
```

The SET VOLUME command in this example causes a rebuild operation to begin on the volume that is mounted on NODE$DBA2:. The /LOG qualifier directs SET VOLUME to display a notification message.

SET WORKING_SET

Redefines the default working set size for the process, or sets an upper limit to which the working set size can be changed by an image that the process executes. Working set limits cannot be set to exceed those defined in the user authorization file (UAF).

FORMAT

SET WORKING_SET

PARAMETERS

None.

DESCRIPTION

The SET WORKING_SET command enables the user to change the working set size within the authorized limits. A process's working set is the collection of pages to which an executing image can refer. Each user is assigned a default working set size to be associated with the process created during login. The maximum size to which any process can increase its working set is defined in the user authorization file.

QUALIFIERS

/ADJUST (default)
/NOADJUST

Enables or disables the system's changing of the process working set.

/EXTENT=n

Specifies the maximum number of pages that can be resident in the working set during image execution.

The extent value must be greater than the minimum working set defined at system generation, and it must be less than or equal to the authorized extent defined in the user authorization file.

If you specify a value greater than the authorized extent, the command sets the working set limit at the maximum authorized value.

/LIMIT=n

Specifies the size to which the working set is to be reduced at image exit.

If you specify a value greater than the current quota, the quota value is also increased.

/LOG
/NOLOG (default)

Determines whether or not confirmation of the SET WORKING_SET command is displayed.

/QUOTA=n

Specifies the maximum number of pages that any image executing in the process context can request. An image can set the working set size for the process by calling the Adjust Working Set Limit ($ADJWSL) system service.

If you specify a quota value that is greater than the authorized quota, the working set quota is set to the authorized quota value.

EXAMPLES

1

```
$ SHOW WORKING_SET
  Working Set      /Limit=  150  /Quota=  700          /Extent= 700
  Adjustment enabled    Authorized Quota=  700  Authorized Extent= 700
$ SET WORKING_SET/QUOTA=1000
%SET-I-NEWLIMS, new working set:   Limit = 150  Quota = 700  Extent = 700
```

The SHOW WORKING_SET command in this example displays the current limit, quota, and extent, as well as the authorized quota and authorized extent. The SET WORKING_SET command attempts to set a quota limiting the maximum number of pages any image can request that is greater than the authorized quota. Note from the response that the quota was not increased.

2

```
$ SHOW WORKING_SET
  Working Set      /Limit=  150  /Quota=  350          /Extent= 350
  Adjustment enabled    Authorized Quota=  350  Authorized Extent= 350
$ SET WORKING_SET/LIMIT=100
%SET-I-NEWLIMS, new working set:   Limit = 100  Quota = 350  Extent = 350
$ SHOW WORKING_SET
  Working Set      /Limit=  100  /Quota=  350          /Extent= 350
  Adjustment enabled    Authorized Quota=  350  Authorized Extent= 350
```

The SET_WORKING SET command in this example sets the working set size for any image in the process to 100.

SHOW

Displays information about the current status of a process, the system, or devices in the system.

FORMAT

SHOW *option*

DESCRIPTION

The SHOW command options are described individually in this manual. Table DCL–15 lists all the SHOW command options.

Table DCL–15 SHOW Command Options

Option	Displays
ACCOUNTING	Items for which accounting is enabled
ACL	The access control list associated with a system object
AUDIT	The security features that are enabled
BROADCAST	Message classes for which broadcast is enabled
CLUSTER	Cluster activity and performance
CPU	Current state of the attached processor
DEFAULT	The current default device and directory
DEVICES	The status of devices in the system
DEVICE/SERVED	The status of devices served by the MSCP server on a VAXcluster
ERROR	The error count for the CPU, memory, and physical devices
INTRUSION	The contents of the breakin database
KEY	Key definitions created by the DEFINE/KEY command
LOGICAL	Current logical name assignments
MAGTAPE	The status and characteristics of a specific magnetic tape device
MEMORY	The availability and usage of memory resources
NETWORK	The availability of network nodes, including the current node
PRINTER	Printer characteristics
PROCESS	Attributes of the current process, including privileges, resource quotas, memory usage, priority, and accounting information
PROTECTION	The current default protection applied to files

Table DCL–15 (Cont.) SHOW Command Options

Option	Displays
QUEUE	Names and types of queues that are available on the system as well as any current jobs belonging to your process
QUEUE/CHARACTERISTICS	Characteristic names and numbers that have been defined for system queues
QUEUE/FORM	Form names and numbers that have been defined for system queues
QUOTA	The current disk quota authorized for and used by a specific user on a specific disk
RMS_DEFAULT	The current default multiblock and multibuffer counts used by RMS for file operations
STATUS	The status of the current job, including accumulated CPU time, open file count, and count of I/O operations
SYMBOL	Current symbol definitions
SYSTEM	A list of all processes in the system
TERMINAL	The device characteristics of a terminal
[DAY]TIME	The current date and time
TRANSLATION	A current logical name assignment
USERS	Information about users currently on the system
WORKING_SET	The current working set size limit and quota

SHOW ACCOUNTING

Displays the activities for which accounting is currently enabled. For a complete description of the Accounting Utility, including information about the ACCOUNTING command, refer to the *VMS Accounting Utility Manual*.

FORMAT

SHOW ACCOUNTING

PARAMETERS

None.

DESCRIPTION

The SHOW ACCOUNTING command displays one or more of the following items for which accounting is currently enabled on your system:

Item	Meaning
PROCESS	Any process termination
INTERACTIVE	Interactive job termination
LOGIN_FAILURE	Login failures
SUBPROCESS	Subprocess termination
DETACHED	Detached job termination
BATCH	Batch job termination
NETWORK	Network job termination
PRINT	All print jobs
MESSAGE	User messages

QUALIFIER

/OUTPUT[=file-spec]
/NOOUTPUT

Specifies the file to which the display is written; by default, the display is written to the current SYS$OUTPUT device.

If you specify /OUTPUT with a partial file specification (for example, specifying only a directory), SHOW is the default file name and LIS the default file type. If you enter a file specification, you may not include any wildcard characters.

If you enter /NOOUTPUT, output is suppressed.

EXAMPLE

```
$ SHOW ACCOUNTING/OUTPUT=ACCOUNTING.SET
```

The SHOW ACCOUNTING command in this example writes the current setting of SET ACCOUNTING to the file ACCOUNTING.SET.

SHOW ACL

Allows you to display the access control list (ACL) of an object.

FORMAT

SHOW ACL *object-name*

PARAMETER

object-name

Specifies the name of the object whose ACL is to be displayed. No wildcard characters are allowed in the object-name specification.

DESCRIPTION

The SHOW ACL command enables you to display the access control list (ACL) of a system object. By default, the SHOW ACL command assumes an object type of file. If the object is any other type, you must include the /OBJECT_TYPE qualifier.

QUALIFIER

/OBJECT_TYPE=type

Defines the object type of the object whose ACL is to be displayed. The following keywords are used to specify the object type:

FILE (default)	The object is a Files–11 disk file.
DEVICE	The object is a device.
SYSTEM_GLOBAL_SECTION	The object is a system global section.
GROUP_GLOBAL_SECTION	The object is a group global section.
QUEUE	The object is a batch or device (terminal, server, or printer) queue.
LOGICAL_NAME_TABLE	The object is a system logical name table.

EXAMPLE

```
$ SHOW ACL/OBJECT_TYPE=DEVICE TTA1
Object type: device,    Object name: VTA1
(IDENTIFIER=[SALES,FRANK],ACCESS=READ)
(IDENTIFIER=[123,321]+NETWORK,ACCESS=NONE)
    .
    .
    .
```

The SHOW ACL command in this example displays the ACL of the device TTA1.

SHOW AUDIT

Displays the security auditing features that are enabled and the events that they report. Also identifies the security auditing failure mode in effect on the system.

Requires the SECURITY privilege.

FORMAT

SHOW AUDIT

PARAMETERS

None.

DESCRIPTION

The SHOW AUDIT command displays the set of features that have been enabled for auditing with the DCL command SET AUDIT. If no auditing has been enabled, the display reports briefly that security alarms are currently disabled. The display is directed to the current SYS$OUTPUT device.

It is useful to check which auditing features are enabled whenever you plan to add or delete features with a subsequent SET AUDIT command.

QUALIFIERS

/ALL (default)

Displays all available auditing information.

/FAILURE_MODE

Displays the failure mode currently in effect on the system.

/OUTPUT[=file-spec]
/NOOUTPUT

Controls where the output of the command is sent. If you do not enter the qualifier, or if you enter /OUTPUT without a file specification, the output is sent to the current process default output stream or device, identified by the logical name SYS$OUTPUT.

If you enter /OUTPUT with a partial file specification (for example, specifying only a directory), SHOW is the default file name and LIS the default file type. If you enter a file specification, it may not include any wildcard characters.

If you enter /NOOUTPUT, output is suppressed.

EXAMPLES

1

```
$ SHOW AUDIT
Security alarms currently disabled
```

The display produced by the SHOW AUDIT command in this example reveals that security auditing is not enabled.

2

```
$ SET AUDIT/ALARM/ENABLE=ALL
$ SHOW AUDIT
Security alarm failure mode is set to:
    WAIT        Processes will wait for resource

Security alarms currently enabled for:
    ACL
    MOUNT
    AUTHORIZATION
    BREAKIN:      (DIALUP,LOCAL,REMOTE,NETWORK,DETACHED)
    LOGIN:        (BATCH,DIALUP,LOCAL,REMOTE,NETWORK,SUBPROCESS,DETACHED)
    LOGFAILURE:   (BATCH,DIALUP,LOCAL,REMOTE,NETWORK,SUBPROCESS,DETACHED)
    LOGOUT:       (BATCH,DIALUP,LOCAL,REMOTE,NETWORK,SUBPROCESS,DETACHED)
    FILE_ACCESS:
        FAILURE: (READ,WRITE,EXECUTE,DELETE,CONTROL)
        SUCCESS: (READ,WRITE,EXECUTE,DELETE,CONTROL)
        SYSPRV:  (READ,WRITE,EXECUTE,DELETE,CONTROL)
        BYPASS:  (READ,WRITE,EXECUTE,DELETE,CONTROL)
        GRPPRV:  (READ,WRITE,EXECUTE,DELETE,CONTROL)
        READALL: (READ,WRITE,EXECUTE,DELETE,CONTROL)
```

The SHOW AUDIT command in this example reveals that security auditing has been enabled to provide alarms for all possible events. The command also reveals that processes attempting to write security alarms when insufficient resources are available will be placed in the MWAIT state to wait for resources.

3

```
$ SHOW AUDIT
Security alarm failure mode is set to:
    WAIT        Processes will wait for resource

Security alarms currently enabled for:
    ACL
    BREAKIN:      (DIALUP,LOCAL,REMOTE,NETWORK,DETACHED)
    FILE_ACCESS:
        FAILURE: (READ,WRITE,EXECUTE,DELETE,CONTROL)
        BYPASS:  (READ,WRITE,EXECUTE,DELETE,CONTROL)
```

The SHOW AUDIT command in this example reveals that all terminals enabled as security operators receive an alarm under the following conditions:

- An access control list (ACL) access requests the alarm
- The system detects a possible breakin attempt
- A file access fails with READ, WRITE, EXECUTE, DELETE, or CONTROL access
- A file access with READ, WRITE, EXECUTE, DELETE, or CONTROL access is gained by means of the BYPASS privilege

4

```
$ SHOW AUDIT/FAILURE_MODE
Security alarm failure mode is set to:
    IGNORE      Alarms will be lost
```

The SHOW AUDIT command in this example shows that the VMS operating system will ignore security alarms when there are insufficient system resources to write the alarm to the operator log file. Terminals enabled as security operators are notified that alarms are being lost.

SHOW AUDIT

5
```
$ SHOW AUDIT
Security alarm failure mode is set to:
    WAIT        Processes will wait for resource

Security alarms currently enabled for:
    BREAKIN:      (DIALUP,LOCAL,REMOTE,NETWORK,DETACHED)
    LOGIN:        (DIALUP)
    LOGOUT:       (DIALUP)
```

The SHOW AUDIT command in this example reveals that the terminals enabled as security operators will receive an alarm whenever the system detects a possible breakin attempt, a dialup at login time, or whenever a dialup connection logs out.

SHOW BROADCAST

Displays the message classes that are currently affected by the SET BROADCAST command.

FORMAT

SHOW BROADCAST

PARAMETERS

None.

DESCRIPTION

The SHOW BROADCAST command tells which classes of messages are being screened from your terminal by the SET BROADCAST command.

QUALIFIER

/OUTPUT[=file-spec]
/NOOUTPUT

Controls where the output of the command is sent. If you do not enter the qualifier, or if you enter /OUTPUT without a file specification, the output is sent to the current process default output stream or device, identified by the logical name SYS$OUTPUT.

If you enter /OUTPUT with a partial file specification (for example, specifying only a directory), SHOW is the default file name and LIS the default file type. If you enter a file specification, it may not include any wildcard characters.

If you enter /NOOUTPUT, output is suppressed.

EXAMPLES

1

```
$ SHOW BROADCAST
Broadcasts are enabled for all classes
```

This example shows the display when all message classes are enabled for broadcast.

2

```
$ SHOW BROADCAST
Broadcasts are currently disabled for:
    MAIL
```

The SHOW BROADCAST display in this example indicates that SET BROADCAST=NOMAIL is in effect.

SHOW BROADCAST

```
3   $ SHOW BROADCAST
    Broadcasts are currently disabled for:
        GENERAL
        PHONE
        MAIL
        QUEUE
        SHUTDOWN
        URGENT
        DCL
        OPCOM
        USER1
        USER2
        USER3
        USER4
        USER5
        USER6
        USER7
        USER8
        USER9
        USER10
        USER11
        USER12
        USER13
        USER14
        USER15
        USER16
```

This example shows the display when SET BROADCAST=NONE is in effect.

SHOW CLUSTER

Invokes the Show Cluster Utility (SHOW CLUSTER) to monitor and display cluster activity and performance. For a complete description of the Show Cluster Utility, including information about the SHOW CLUSTER command, see the *VMS Show Cluster Utility Manual*.

FORMAT SHOW CLUSTER

SHOW CPU

Displays the current state of the processors in a VMS multiprocessing system.

Applies only to VMS multiprocessing systems. Requires change mode to kernel (CMKRNL) privilege.

FORMAT

SHOW CPU *[cpu-id,...]*

PARAMETER

cpu-id

Decimal value representing the identity of a processor in a multiprocessing system. In a VAX 8300 system, for instance, the CPU ID is the VAXBI node number of the processor; in a VAX 8800, the CPU ID of the left processor is 1 and that of the right processor is 0.

DESCRIPTION

The SHOW CPU command displays information about the status, characteristics, and capabilities of the processors active in and available to a VMS multiprocessing system.

You identify the processors to be displayed by using either the /ACTIVE qualifier, the /ALL qualifier, a CPU ID, or list of CPU IDs. If you specify none of these, the SHOW CPU command uses the /ALL qualifier by default.

You identify the type of information to be displayed by using the /BRIEF, /FULL, and /SUMMARY qualifiers. If you specify neither the /SUMMARY, /BRIEF, nor /FULL qualifier, SHOW CPU assumes the /BRIEF qualifier by default. However, if you likewise do not identify a processor or processors as the object of a command, SHOW CPU assumes a default of SHOW/ALL /SUMMARY.

QUALIFIERS

/ACTIVE

Selects as the subject of the display only those processors that are members of the system's active set.

/ALL

Selects all configured processors, active and inactive, as the subject of the display.

/BRIEF

Produces information from the summary display and also lists the current CPU state and current process (if any) for each processor in the configuration.

/FULL

Produces information from the summary display. The /FULL qualifier also lists the current CPU state, current process (if any), revision levels, and capabilities for each configured processor. It indicates which processes can execute only on certain processors in the configuration. In addition, if one or more uniprocessing drivers are present in the system, the /FULL qualifier lists them by name.

/SUMMARY

Produces a display listing the processors in the VMS multiprocessing system, indicating which is the primary, which are configured, and which are active. The /SUMMARY qualifier also indicates the minimum revision levels required for processors in the system, which VMS synchronization image has been loaded into the operating system, and whether multiprocessing is enabled. If the presence of one or more uniprocessing drivers in the system prohibits the enabling of multiprocessing, the SHOW CPU command displays a warning message.

EXAMPLES

1

```
$ SHOW CPU
<
SOWHAT, A VAX 8800
Multiprocessing is ENABLED. Full checking synchronization image loaded.
Minimum multiprocessing revision levels: CPU = 0 uCODE = 0 UWCS = 0.

PRIMARY CPU = 01
Active CPUs:      00 01
Configured CPUs:  00 01
```

The SHOW CPU command in this example produces a configuration summary of all configured processors in the VAX 8800 system SOWHAT. The primary processor is CPU 01, and all configured processors are active.

2

```
$ SHOW CPU/BRIEF

SOWHAT, A VAX 8800
Multiprocessing is ENABLED. Full checking synchronization image loaded.
Minimum multiprocessing revision levels: CPU = 0 uCODE = 0 UWCS = 0.

PRIMARY CPU = 01

CPU 00 is in RUN state
Current Process: AIREGIN          PID = 4A8001E5

CPU 01 is in RUN state
Current Process: ***None***
```

The SHOW CPU/BRIEF command in this example produces a configuration summary of the VAX 8800 system SOWHAT and also indicates that its two processors are in the RUN state. Only CPU 00 has a current process.

3

```
$  SHOW CPU/FULL

SOWHAT, A VAX 8800
Multiprocessing is ENABLED. Full checking synchronization image loaded.
Minimum multiprocessing revision levels: CPU = 0 uCODE = 0 UWCS = 0.

PRIMARY CPU = 01

CPU 00 is in RUN state
Current Process: AIREGIN          PID = 4A8001E5
Revision levels: CPU = 0  uCODE = 0  UWCS = 0.
Capabilities of this CPU:
    *** None ***
Processes which can only execute on this CPU:
    *** None ***
```

```
CPU 01 is in RUN state
Current Process: *** None ***
Revision levels: CPU = 0  uCODE = 0  UWCS = 0.
Capabilities of this CPU:
        PRIMARY TIMEKEEPER
Processes which can only execute on this CPU:
        CONFIGURE         PID = 4A80010C  Reason = PRIMARY Capability
```

The SHOW CPU/FULL command in this example produces a configuration summary of the VAX 8800 system SOWHAT, indicating that each processor is in the RUN state. It also shows that CPU 01 has primary and timekeeper capabilities. There is one process, CONFIGURE, in the system that can execute only on CPU 01 because only that processor has the primary capability.

4

```
$ SHOW CPU

OLEO, A VAX 8300
Multiprocessing is DISABLED. Full checking synchronization image loaded.
Minimum multiprocessing revision levels: CPU = 0 uCODE = 0 UWCS = 0.
*** Loaded unmodified device drivers prevent multiprocessor operation.***
PRIMARY CPU = 02
Active CPUs:      02
Configured CPUs:  02 08
```

The SHOW CPU command in this example produces a configuration summary of all configured processors in the VAX 8300 system OLEO. The primary processor is CPU 02. Multiprocessing cannot be enabled, and the secondary processor cannot be booted because a uniprocessing device driver is present in the system.

5

```
$ SHOW CPU/FULL

OLEO, A VAX 8300
Multiprocessing is DISABLED. MULTIPROCESSING Sysgen parameter = 02
Minimum multiprocessing revision levels -- CPU: 0 uCODE: 0 UWCS: 21.
PRIMARY CPU = 01
*** Loaded unmodified device drivers prevent multiprocessor operation.***
    RBDRIVER

CPU 02 is in RUN state
Current Process: Koko               PID = 2A6001E3
Revision levels: CPU: 0 uCODE: 0 UWCS: 0.
Capabilities of this CPU:
        PRIMARY TIMEKEEPER
Processes which can only execute on this CPU:
        CONFIGURE         PID = 2A40010B  Reason = PRIMARY Capability

CPU 07 is in INIT state
Current Process: *** None ***
Revision levels: CPU: 0 uCODE: 0 UWCS: 0.
Capabilities of this CPU:
    *** None ***
Processes which can only execute on this CPU:
    *** None ***
```

The SHOW CPU/FULL command in this example produces a configuration summary of the VAX 8300 system OLEO, indicating that only CPU 02, the primary is active and in the RUN state. It also shows that there is a uniprocessing driver loaded in the system, thus preventing the system from being enabled as a multiprocessor.

SHOW DEFAULT

Displays the current default device and directory.

FORMAT SHOW DEFAULT

DESCRIPTION

The SHOW DEFAULT command displays the current device and directory names, along with any equivalence strings.

The default disk and directory are established in the user authorization file. You can change these defaults during a terminal session or in a batch job with the SET DEFAULT command, or by reassigning the logical name SYS$DISK.

EXAMPLES

1

```
$ SHOW DEFAULT
  DISK1:[ALPHA]
$ SET DEFAULT DISK5:[HIGGINS.SOURCES]
$ SHOW DEFAULT
  DISK5:[HIGGINS.SOURCES]
```

The SHOW DEFAULT command in this example displays the current default device and directory names. The SET DEFAULT command changes these defaults, and the next SHOW DEFAULT command displays the new default device and directory.

2

```
$ SET DEFAULT NOSUCH:[NOWAY]
$ SHOW DEFAULT
 NOSUCH:[NOWAY]
 %DCL-I-INVDEF, NOSUCH:[NOWAY] does not exist
```

In this example, the default has been set to a nonexistent device and directory. An error message is displayed.

3

```
$ DEFINE/TRANSLATION_ATTRIBUTES=CONCEALED XYZ WORK:[INVOICES.]
$ SET DEFAULT XYZ:[SALES]
$ SHOW DEFAULT
XYZ:[SALES]
```

In this example, a logical name, XYZ, is defined to represent WORK:[INVOICES]. The TRANSLATION_ATTRIBUTES=CONCEALED qualifier tells the system not to display the translation of XYZ in file specifications. Thus, the SHOW DEFAULT command displays the logical name XYZ and not its translation.

4

```
$ SET DEFAULT WORK:[BLUE]
$ SHOW DEFAULT
 WORK:[BLUE]
$ DEFINE FOO WORK:[BLUE.TEMP1],WORK:[BLUE.TEMP2]
$ SET DEFAULT FOO
$ SHOW DEFAULT
 FOO:[BLUE]
 =    WORK:[BLUE.TEMP1]
 =    WORK:[BLUE.TEMP2]
```

In this example, the logical name FOO is defined as a search list containing the directories [BLUE.TEMP1] and [BLUE.TEMP2] both on device WORK. The SET DEFAULT command equates this search list logical name with the logical name SYS$DISK. The subsequent SHOW DEFAULT command displays the search list logical name along with its equivalence strings.

Because the directory field has not been explicitly specified, the original [BLUE] directory remains in effect as the current default directory. However, unless the current default directory syntax ([]) is explicity used, all file references are to those directories contained in the search list.

SHOW DEVICES

Displays the status of a device on the system.

See the qualifier descriptions for restrictions.

FORMAT

SHOW DEVICES *[device-name[:]]*

PARAMETER

device-name[:]

Specifies the name of a device for which information is to be displayed. You can specify a complete device name or only a portion of a device name. The SHOW DEVICES command provides defaults for nonspecified portions of device names, as follows:

- If you truncate a device name (for example, if you specify D), the command lists information about all devices whose device names begin with what you entered (in this case, D).
- If you omit a controller designation, the SHOW DEVICES command lists all devices on all controllers with the specified unit number.
- If you omit a unit number, the SHOW DEVICES command lists all devices on the specified controller.

If you enter the SHOW DEVICES command and specify neither a device name parameter nor any qualifier, the command provides a brief listing of characteristics of all devices on the system (with the exception of mailbox devices). To obtain information about a specific device or generic class of devices, specify a device name.

Use the /ALLOCATED qualifier for a list of devices that are currently allocated to processes; use the /MOUNTED qualifier for a list of the mounted devices.

Note that the /FILES qualifier does not support defaults for nonspecified portions of device names; you must supply a complete device specification.

DESCRIPTION

When you enter the SHOW DEVICES command without specifying a device or using any qualifier, information about all devices on the system is displayed. If you specify a device name, the SHOW DEVICES command displays information about that device. If you use certain qualifiers with SHOW DEVICES, information is displayed about those devices that currently have volumes mounted or that have been allocated to processes, or both.

The device name displayed by the system uses the format *ddcu* where *dd* is the device code, *c* is the controller designation, and *u* is the unit number. If the system is part of a VAXcluster that is running with hierarchical storage controllers (HSCs), the device name will include the node name using the format *node$ddcu* (where *node* refers to the node name).

QUALIFIERS

/ALLOCATED

Displays all devices currently allocated to processes.

If you specify a device name, the characteristics of only that device are displayed. If the device is not currently allocated, the command displays a message indicating that there is no such device. If you specify a generic device name, the characteristics of all allocated devices of that type are displayed.

/BRIEF (default)

Displays brief information about the specified devices.

/FILES

Requires SYSPRV or BYPASS privileges to list read-protected files.

Displays a list of the names of all files open on a volume and their associated process name and process identification (PID). The specified device must be a mounted Files–11 volume. If the specified volume is a multivolume set, the files on each volume in the set are listed.

Note that the SHOW DEVICES/FILES command does not support defaults for nonspecified portions of device names. You must supply a complete device specification when using the /FILES qualifier.

If the /SYSTEM qualifier is also specified, only the names of installed files and files opened by the system are displayed. Files opened by the system are those that have been opened without the use of an ancillary control process (ACP), such as INDEXF.SYS and QUOTA.SYS, as well as system files such as JBCSYSQUE.EXE and SYSMSG.EXE.

If the /NOSYSTEM qualifier is specified, only those files opened by processes are displayed. To list files opened by a process in your group, your process must have at least GROUP privilege. If the process is not in your group, you need WORLD privilege.

If neither the /SYSTEM nor /NOSYSTEM qualifier is specified, the names of all files currently opened on the system are displayed.

If a file is read-protected from your UIC, the "No privilege" message is displayed instead of the file name. You must have SYSPRV privilege or BYPASS privilege to display the file name.

A space in place of a file name represents a workfile (such as a temporary edit file) not entered in any directory. To display temporary file names, you must have BYPASS privilege in addition to GROUP or WORLD privilege.

Do not use the /FILES qualifier with the /ALLOCATED, /BRIEF, /FULL, or /MOUNTED qualifiers. The functions of /FILES and these qualifiers are mutually exclusive.

/FULL

Displays a complete list of information about the devices.

/MOUNTED

Displays all devices that currently have volumes mounted on them.

If you specify a device name, only the characteristics of that device are displayed. However, if the device is not currently mounted, the command issues a message indicating there is no such device. If you specify a generic device name, the characteristics of all such devices that currently have volumes mounted are displayed.

/OUTPUT[=file-spec]
/NOOUTPUT

Controls where the output of the command is sent. If you do not enter the qualifier, or if you enter /OUTPUT without a file specification, the output is sent to the current process default output stream or device, identified by the logical name SYS$OUTPUT.

If you enter /OUTPUT with a partial file specification (for example, specifying only a directory), SHOW is the default file name and LIS the default file type. If you enter a file specification, it may not include any wildcard characters.

If you enter /NOOUTPUT, output is suppressed.

/SYSTEM
/NOSYSTEM

Controls whether the names of installed files and files opened by the system are displayed. Files opened by the system are those that have been opened without the use of an ancillary control process (ACP), such as INDEXF.SYS and QUOTA.SYS.

If you specify /NOSYSTEM with the /FILES qualifier, only files opened by processes are displayed. If you omit both /SYSTEM and /NOSYSTEM and specify the /FILES qualifier, the names of all files currently opened on the system are displayed.

You can use this qualifier only with the /FILES qualifier. See the description of the /FILES qualifier for more details.

/WINDOWS

Displays the window count and total size of all windows for files open on a volume. The file name and related process name and process identification (PID) are also displayed. The letter C in a display indicates that the file is open with "cathedral windows" (segmented windows).

EXAMPLES

1

```
$ SHOW DEVICES
Device                  Device      Err.    Volume         Free  Trans Mount
 Name                   Status      Count    Label        Blocks Count Count
DBA0:                   Online mnt      0  VMS            47088   115     1
DBA1:                   Online mnt      0  USERPACK1      45216     2     1
DBA2:                   Online mnt      3  DOCUMENT        8068    20     1
DBA5:                   Online mnt      0  MASTERP        28668     1     1
DBA6:                   Online          0
DBA7:                   Online mnt      0  PROJECT       110547     1     1
DMA0:                   Online          0
DLA0:                   Online          0
DYA0:                   Online          0
DYA1:                   Online          0
DRA3:                   Online mnt      0  RES26APR       29317     1     1
```

In this example, the SHOW DEVICES command displays the following information for each device on the system:

- Device name.
- Device status and characteristics. (Status indicates whether the device is online; characteristics indicate whether the device is allocated, spooled, has a volume mounted on it, or has a foreign volume mounted on it.)
- Error count.
- Volume label (for disk and tape volumes only).
- Number of free blocks on the volume.
- Transaction count.
- Number of mount requests issued for the volume (disk devices only).

2
```
$ SHOW DEVICES/FULL DMA0
Disk NODE1$DMA0:, device type RK07, is online, allocated, mounted,
error logging enabled
 Error count                    0  Operations completed            1257
 Owner UIC                [1,4]  Owner process name             VANNOY
 Owner process ID      202000C8  Dev Prot  S:RWED,O:RWED,G:RWED,W:RWED
 Reference count               2  Default buffer size                512
 Volume label      JAKE_X239      Relative volume no.                  0
 Cluster size                  1  Transaction count                    2
 Free blocks                3741  Maximum files allowed            13447
 Extend quantity               5  Mount count                          1
 Volume status           Process  ACP process name              DMA0BACP
 File ID cache size           64  Extent cache size                   64
 Quota cache size             64
Volume is subject to mount verification, file high-water marking
```

In this example, the SHOW DEVICES command requests a full listing of the status of the RK07 device DMA0. The device is located on NODE1 in a VAXcluster.

```
3  $ SHOW DEVICES/FULL NODE2$
Disk NODE2$DUA0:, device type RA81, is online, mounted,
error logging enabled
 Error count                    0  Operations completed           24195
 Owner UIC              [11,177]  Owner process name
 Owner process ID       20200000  Dev Prot  S:RWED,O:RWED,G:RWED,W:RWED
 Reference count              16  Default buffer size               512
 Volume label       VMSDOCLIB     Relative volume no.                 0
 Cluster size                 3  Transaction count                  17
 Free blocks             525447  Maximum files allowed          111384
 Extend quantity              5  Mount count                         1
 Volume status           System  ACP process name
 Caching disabled
Volume is subject to mount verification, file high-water marking

Disk NODE2$DUA1:, device type RA81, is online, error logging enabled
 Error count                    0  Operations completed               0
 Owner UIC                [0,0]  Owner process name
 Owner process ID       20200000  Dev Prot  S:RWED,O:RWED,G:RWED,W:RWED
 Reference count                0  Default buffer size               512
    .
    .
    .
```

In this example, the user requested a full display of information about each device on NODE2 in the VAXcluster. Information is shown here only for the first two devices: a mounted device and a device that is not mounted.

SHOW DEVICES/SERVED

Displays information on devices served by the MSCP server on this node. The /SERVED qualifier is required.

FORMAT

SHOW DEVICES/SERVED

DESCRIPTION

The SHOW DEVICES/SERVED command displays information about the MSCP server and the devices it serves. This information is mostly used by system managers.

QUALIFIERS

/ALL

This qualifier displays the information displayed by all of the qualifiers listed below except the /OUTPUT qualifier.

/COUNT

Displays the number of transfer operations completed, sorted by the size of the transfers, and the number of MSCP operations that have taken place since the MSCP server was started.

/HOST

Displays the names of the processors that have MSCP-served devices on line. SYSGEN's MSCP/HOST command determines how many hosts in the cluster can connect to the MSCP server at one time.

/OUTPUT=[filespec]

Redirects output from your terminal to the specified file. If you do not specify a file, or if you do not use this qualifier, output is sent to SYS$OUTPUT.

/RESOURCE

Displays information on the resources available to the MSCP server for use in processing I/O requests for the devices it serves. You make these resources available to the MSCP server when you use SYSGEN's MSCP command to start the MSCP server and use the qualifiers listed in the following table:

Qualifier	Item Specified
/BUFFER	The amount of buffer space available to the MSCP server
/FRACTION	The maximum size, in pages, of the buffer granted to an I/O request; for transfers of more data than will fit a buffer of the size specified by this qualifier, several CI transfers are needed
/SMALL	The minimum size, in pages, of the buffer that the MSCP server can grant to an I/O request; if less than this amount of buffer space is available, the I/O request must wait until at least this much buffer space becomes available; when this much space becomes available, the MSCP server grants the request a buffer
/PACKETS	The number of I/O-request packets (CDRPs) available to the MSCP server for processing I/O requests

EXAMPLES

1

```
SHOW DEVICES/SERVED
MSCP Served Devices on BOSTON 31-DEC-1988 12:34:56.78
                                          Queue Requests
Device:          Status      Total Size     Current    Max     Hosts
    2$DBA0        AVAIL          340670           0      0         0
    2$DMA1       ONLINE           53790           0      0         2
    2$DMA0      OFFLINE           53790           0      0         0
```

This example shows the output generated by the command SHOW DEVICES /SERVED. The first column in the display shows the names of the devices that are served by the MSCP server. The second column shows the status of the devices. The third column shows the size, in blocks, of the device.

The *Queue Requests* column shows the number of I/O requests currently awaiting processing by that device and the maximum number of I/O requests that have ever been concurrently awaiting processing by that device. The last column in the display shows the number of hosts that have the device on line.

2

```
$ SHOW DEVICES/SERVED/COUNT
       MSCP Served Devices on BOSTON 31-DEC-1988 12:34:58.82
Request Count:
       0-7:        1        32-39:        8        88-103:        7
      8-15:        0        40-55:        0       104-127:        0
     16-23:        0        56-71:        8
     24-31:        0        72-87:       20
Operations Count:
     ABORT            0     ERASE            0     READ             0
     ACCESS           0     FLUSH            0     REPLACE          0
     AVAILABLE        0     GET CMD STS      0     SET CTL CHR     10
     CMP CTL DAT      0     GET UNT STS   2799     SET UNT CHR      0
     CMP HST DAT      0     ONLINE           0     WRITE            0
   Total           2809
```

This example shows the information displayed by the SHOW DEVICES /SERVED/COUNT command. The numbers to the left of the colon, separated by a hyphen, are the size, in pages, of the requests. The numbers to the right of the colon are the number of requests of that size that have been processed by the MSCP server.

The section of the display headed by the label *Operations Count* shows the number of times the MSCP server has performed the MSCP operations listed. In the example, this MSCP server has performed 10 set-controller-characteristics (SET CTL CHR) operations, but has performed no set-unit-characteristics (SET UNT CHR) operations.

3

```
$ SHOW DEVICES/SERVED/RESOURCE
       MSCP Served Devices on BOSTON 31-DEC-1988 12:34:58.82
                                                        Fragment Size
Resources:          Total      Free      In Use     Minimum    Maximum
    Buffer area:       64        64           0           4         32
    I/O Packets:      144       144
                  Current   Maximum
    Buffer wait:        0         0
```

This example shows the information displayed by the SHOW DEVICES /RESOURCE command. The *Total* column shows the total number of pages in the buffer area and in the number of I/O-request packets set aside for use by the MSCP server. The *Free* column shows the number of pages in the buffer and the number of I/O-request packets that are available for use.

The *In Use* column shows the number of pages within the buffer area that are in use. The columns labeled *Fragment Size* show the minimum and the maximum size, in pages, of a buffer that an I/O request can obtain from the buffer area.

The line labeled *Buffer wait* shows the number of I/O requests that are currently waiting for buffer space to become available for their use, and the maximum number of I/O requests that have concurrently waited to obtain a buffer.

4

```
$ SHOW DEVICES/SERVED/HOST
       MSCP Served Devices on BOSTON 31-DEC-1988 12:34:58.82
                                          Queue Requests
Host:                Time of Connection    Current    Max   Devices
    HARVEY       30-DEC-1988 12:57:39.90          0      1         2
    DOC          30-DEC-1988 22:02:10.25          0      1         0
    GRUMPY       30-DEC-1988 22:02:10.25          0      0         1
    SLEEPY       30-DEC-1988 22:02:11.75          0      1         0
```

This example shows the information displayed by the SHOW DEVICES /SERVED/HOST command. The first column contains the names of the hosts that have class drivers connected to the MSCP server. The next column contains the times at which these connections were made.

The columns under the heading *Queue Requests* show the number of requests the MSCP server has currently outstanding for I/O activity on the devices it serves, the maximum number of such requests that have been outstanding at one time, and the number of MSCP-served devices that the listed hosts have on line.

SHOW ENTRY

Displays information about a user's batch and print jobs or about specific job entries.

Requires GROUP privilege to display all jobs in your group. Requires OPER privilege to display all jobs in all groups.

FORMAT

SHOW ENTRY *[entry-number,...]*

PARAMETER

[entry-number,...]

Specifies the entry number of the job you want displayed. If no entry number is specified, all your jobs (or those owned by the user specified with the /USER_NAME qualifier) are displayed.

DESCRIPTION

The SHOW ENTRY command displays information about a user's batch and print jobs or about specific job entries.

The display lists each entry's current status as well as its attributes. These attributes are the job name, owner, entry number, job status, and queue name. You may obtain this information about your print and batch jobs or another user's jobs on any queue.

If you are interested solely in the status of your batch or print jobs, the SHOW ENTRY command produces a more efficient display than the SHOW QUEUE command. Because the SHOW QUEUE command displays all available queues and the jobs they contain, you must scan the display to locate your jobs. By contrast, SHOW ENTRY displays the status of only your jobs (or those owned by the user you specify with the /USER_NAME qualifier).

QUALIFIERS

/BATCH

Selects batch jobs for display. If /USER_NAME is not specified, information about your jobs is displayed.

/BRIEF (default)

Displays the following information for each job: job name, user name, entry number, job size in blocks (for print jobs), status, queue name, and queue type. The /FULL and /FILES qualifiers override /BRIEF.

Specify the /FULL qualifier to obtain more job information.

/BY_JOB_STATUS[=(keyword,...)]

Selects for display only those jobs with the specified status. Specify the status with one or more of the following keywords:

EXECUTING	Requests the display of currently executing jobs.
HOLDING	Requests the display of jobs on hold. Holding status indicates that the job is being held in the queue indefinitely.
PENDING	Requests the display of jobs with pending status. Pending status indicates that the job is waiting its turn to execute.
RETAINED	Requests the display of jobs retained in the queue after execution. Retained status indicates that the job has completed but remains in the queue. For example, a job may be retained in the queue if there was an error during its execution.
TIMED_RELEASE	Requests the display of jobs on hold until a specified time. Timed release status indicates that the job is being held in the queue for execution at a future time.

If no keyword is specified, /BY_JOB_STATUS displays the status of all jobs.

If /USER_NAME is not specified, information about your job is displayed.

/DEVICE[=(keyword,...)]

Selects for display only those print jobs in the queue types specified. Specify the queue type with one or more of the following keywords:

PRINTER	Requests the display of jobs in print queues.
SERVER	Requests the display of jobs in server queues.
TERMINAL	Requests the display of jobs in terminal queues.

If no keyword is specified, /DEVICE displays all printer, terminal, and server queues.

If /USER_NAME is not specified, information about your jobs is displayed.

/FILES

Adds to the default display the list of full file specifications for each file in each job.

/FULL

Displays the following information for each job: job name, user name, entry number, job status, full file specification associated with each job, date and time of submission, settings specified for the job, queue name, and queue type.

The /FULL qualifier overrides the default brief listing format.

/GENERIC

Selects for display only those jobs contained in generic queues. A generic queue holds jobs of a particular type (for example, batch or line printer jobs) and directs them to execution queues for processing.

If /USER_NAME is not specified, information about your jobs is displayed.

/OUTPUT[=filespec]
/NOOUTPUT

Controls where the output of the SHOW ENTRY command is sent. By default, the output is sent to the current SYS$OUTPUT device (usually your terminal). To send the output to a file, use the /OUTPUT qualifier followed by a file specification.

The file specification may not include any wildcard characters. If you enter a partial file specification (for example, specifying only a directory), SHOW is the default file name and LIS is the default file type.

If you enter /NOOUTPUT, output is suppressed.

/USER_NAME=username

Selects for display those jobs owned by the specified user. If /USER_NAME is not specified, information about your jobs is displayed.

EXAMPLES

1

```
$ SHOW ENTRY/DEVICE=(PRINTER,TERMINAL)
Jobname         Username    Entry    Blocks   Status
-------         --------    -----    ------   ------
FORECAST        JONES         422        12   Printing
  On printer queue LN01$PRINT

MANAGER         JONES         431         4   Printing
  On terminal queue LQ$PRINT
```

In this example, SHOW ENTRY produces a display of your current job entries on all printer and terminal queues.

2

```
$ SHOW ENTRY/USER_NAME=MACDUFF/FULL
Jobname         Username    Entry    Blocks    Status
-------         --------    -----    ------    ------
STAFF           MACDUFF       625       112    Pending
  On printer queue LN01$PRINT
  Submitted  5-FEB-1988  12:14 /FORM=LN01$PORTRAIT (stock=DEFAULT) /NOTIFY /PRIORITY=100
    _DBA1:[MACDUFF]STAFF.DIS;3  (pending)

MEMO            MACDUFF       629        94    Printing at block 37
  One printer queue LINE$PRINT
  Submitted  4-FEB-1988  12:16  /FORM=DEFAULT  /NOTIFY  /PRIORITY=100
    _DBA1:[MACDUFF.DAILY]MEMO.TXT;2  (printing copy 2)  /COPIES=2 /NOFEED  /PASSALL
```

In this example, DUNCAN has requested a display of the current entries owned by MACDUFF on all queues. The /FULL qualifier lists the submission information, the full file specification, and the current settings for both the job and the queue.

Note that entry 629 specified the /COPIES, /NOFEED, and /PASSALL qualifiers to the PRINT command. /NOFEED suppresses automatic form feeds. /PASSALL suppresses formatting (including form feeds and carriage return characters) performed by the print symbiont.

3

```
$ SUBMIT ASSIGNMENTS.COM
Job ASSIGNMENTS.COM (queue SYS$BATCH, entry 199) pending)
   .
   .
   .
$ SHOW ENTRY 199
  Jobname           Username    Entry    Status
  -------           --------    -----    ------
  ASSIGNMENTS       JONES         199    Executing
    On batch queue WRITER_BATCH
```

In this example, JONES submits ASSIGNMENTS.COM for batch processing. Because JONES does not specify a specific queue, the job is entered into the generic SYS$BATCH queue to await processing. After performing other tasks, JONES checks the status of her job and sees that her file is now executing. Note that the job entry migrated from a generic to an execution batch queue and that JONES was able to check the status of her job without knowing the specific batch queue name. If she did not specify an entry number, all her jobs would have been displayed.

SHOW ERROR

Displays the error count for all devices with error counts greater than 0.

FORMAT

SHOW ERROR

PARAMETERS

None.

QUALIFIERS

/FULL

Displays the error count for all devices, including those with no errors. (The error count is either 0 or a number greater than 0.)

/OUTPUT[=file-spec]
/OUTPUT=SYS$OUTPUT (default)

Specifies the file to which the display is written. By default, the display is written to the current SYS$OUTPUT device.

If you enter /OUTPUT with a partial file specification (for example, specifying only a directory), SHOW is the default file name and LIS the default file type. If you enter a file specification, it may not include any wildcard characters.

If you enter /NOOUTPUT, output is suppressed.

EXAMPLE

```
$ SHOW ERROR
```

Displays the error count for all devices with error counts greater than 0:

```
Device  Error Count
CPU         2
MEMORY      1
DBB1        9
```

SHOW INTRUSION

Displays the contents of the break-in database.

Requires the CMKRNL and SECURITY privileges.

FORMAT SHOW INTRUSION

DESCRIPTION

The VMS operating system stores information in the break-in database about login failures that originate from a specific source and that result from any number of failure types (invalid password, account expired, unknown user name). A security manager can identify possible break-in attempts by using the SHOW INTRUSION command to display the contents of the break-in database.

The entries in the break-in database have the following format:

```
Intrusion      Type      Count      Expiration      Source
```

The information provided in the fields in each entry is as follows:

Field	Description
Intrusion	Class of intrusion. The type of evasive action that the VMS operating system takes depends on the class of intrusion.
Type	Severity of intrusion as defined by the threshold count for login failures.
Count	Number of login failures associated with a particular source.
Expiration	Absolute time at which a login failure is no longer counted by VMS. The SYSGEN parameter, LGI_BRK_TMO, controls how long the VMS operating system keeps track of a login failure.
Source	Origin of the login failure. The information provided in this field depends on the class of intrusion.

In the break-in database, the operating system classifies login failures according to their source. The four classes of system intrusion are as follows:

Intrusion Class	Description
TERMINAL	Login failure originating from one terminal.
TERM_USER	Login failure originating from one terminal, using a valid user name.
NETWORK	Login failure originating from a remote node using a valid user name.
USERNAME	Login failure attempting to create a detached process.

The class of intrusion determines the type of information presented in the source field of the entry. Information appears in the source field in one of the following formats:

Intrusion Class	Format of Source Field
TERMINAL	terminal:
TERM_USER	terminal:user name
NETWORK	node::user name
USERNAME	user name

The type of evasive action that a security manager can take is based on the type of information provided. See the *Guide to VMS System Security* for details on how to use this information.

The break-in database contains two levels of intrusion entries: suspect and intruder. The severity level of an entry is displayed in the type field of the entry. When a login failure associated with a particular source occurs, the VMS operating system classifies the login failure as suspect. Each succeeding login failure from the same source is counted. The login failure count is displayed in the count field of the entry. The absolute time at which the login failure ceases to be counted is displayed in the expiration field of the entry. When the number of login failures exceeds the number specified by the SYSGEN parameter, LGI_BRK_TERM, the break-in entry is classified as intruder.

When a break-in entry is promoted to intruder, the VMS operating system takes evasive action by blocking all login attempts from that particular source.

The duration of the evasive action is determined by the SYSGEN parameter LGI_HID_TIM. The absolute time at which the evasive action ends is displayed in the expiration field of the entry.

For information on break-in detection, prevention, and evasive actions, see the *Guide to VMS System Security*.

If you determine that an entry in the break-in database resulted from a user error and not a break-in attempt, you can remove an entry from the break-in database with the DELETE/INTRUSION command. See the DELETE /INTRUSION command for more details.

QUALIFIERS

/OUTPUT[=file-spec]

Directs the output from the SHOW INTRUSION command to the file specified with the qualifier. By default, output from the command is displayed to SYS$OUTPUT.

/TYPE=keyword

Selects the type of information from the break-in database that is displayed. The valid keywords are as follows:

SHOW INTRUSION

ALL	All break-in entries. By default, all entries are displayed.
SUSPECT	Break-in entries for login failures that have occurred but have not yet passed the threshold necessary to be identified as intruder.
INTRUDER	Break-in entries for which the login failure rate was high enough to warrant evasive action.

EXAMPLES

1
```
$ SHOW INTRUSION/OUTPUT=INTRUDER.LIS
```

The SHOW INTRUSION command in this example writes all the entries currently in the break-in database to the file INTRUDER.LIS.

2
```
$ SHOW INTRUSION/TYPE=INTRUDER
Intrusion   Type       Count    Expiration    Source
TERMINAL    INTRUDER     9      10:29:39.16   _LTA23:
NETWORK     INTRUDER     7      10:47:53.12   STAR::HAMM
```

In this example, the SHOW INTRUSION command displays all intruder entries currently in the break-in database.

SHOW KEY

Displays the key definitions created with the DEFINE/KEY command.

FORMAT

SHOW KEY *[key-name]*

PARAMETER

key-name

The name of the key whose definition you want displayed. See the DEFINE /KEY command for a list of valid key names.

DESCRIPTION

After you have defined keypad keys using the DEFINE/KEY command, you can use the SHOW KEY command to refresh your memory about a key definition. You can also use the SHOW KEY command with the /DIRECTORY qualifier to find out the names of all the states in which you have created key definitions. Use the SET KEY command to change key states.

QUALIFIERS

/ALL

Displays all key definitions in the current state (or the state specified with the /STATE qualifier). If you use the /ALL qualifier, do not specify a key name.

/BRIEF (default)
/NOBRIEF

Displays only the key definition and state. The /BRIEF and /NOFULL qualifiers are equivalent.

/DIRECTORY

Displays the names of all states for which keys have been defined. If you have not specified a state with a key definition, the SHOW KEY/DIRECTORY command displays DEFAULT for the state.

You cannot use the /DIRECTORY qualifier with any of the other SHOW KEY qualifiers.

/FULL
/NOFULL (default)

Displays all qualifiers associated with a definition. By default, only the state of the definition and the definition itself are displayed. The /FULL and /NOBRIEF qualifiers are equivalent.

/STATE=(state-name[,...])
/NOSTATE

Displays the key definitions for the specified state. If you specify only one state name, you can omit the parentheses. State names can be any appropriate alphanumeric string. State names are created with the DEFINE /KEY command.

If you omit the /STATE qualifier or use /NOSTATE, key definitions in the current state are displayed.

EXAMPLE

```
$ DEFINE/KEY/TERMINATE PF1 "ATTACH GEORGE"
%DCL-I-DEFKEY, DEFAULT key PF1 has been defined
$ SHOW KEY PF1
DEFAULT keypad definitions:
  PF1 = "ATTACH GEORGE"
$ SHOW KEY/FULL PF1
DEFAULT keypad definitions:
  PF1 = "ATTACH GEORGE"  (noecho,terminate,noerase,nolock)
```

The SHOW KEY command in this example displays both the definition and the state for the PF1 key. This is the default display. The SHOW KEY/FULL command displays all qualifiers associated with the key definition.

SHOW LOGICAL

Displays translations, the level of translation, and the logical name table for a specified logical name. The SHOW LOGICAL command performs iterative translations.

Requires READ (R) access to the table in which a logical name is cataloged to display information about the logical name.

FORMAT

SHOW LOGICAL *[logical-name[:][,...]]*

PARAMETER

logical-name[:][,...]

Specifies one or more logical names whose translations you want to display. The asterisk (*) and percent (%) wildcard characters are allowed. However, if a wildcard character is used, iterative translation is not done.

The logical name is translated iteratively up to a number of times determined by the system (from 9 to 11). That is, translations are examined to see if they are also logical names.

DESCRIPTION

The SHOW LOGICAL command displays logical names. If you specify a logical name, its translations are displayed. If you do not specify a logical name, all the logical names in the tables defined by LNM$DCL_LOGICAL are displayed.

You can specify the tables you want to search. If you do not specify a table, SHOW LOGICAL searches the tables specified by LNM$DCL_LOGICAL.

LNM$DCL_LOGICAL contains the list of logical name tables and the order in which they are searched. Unless LNM$DCL_LOGICAL has been redefined, the process, job, group, and system tables are searched, in that order. (To see how LNM$DCL_LOGICAL is defined for your process, enter the command SHOW LOGICAL/TABLE=LNM$DIRECTORIES LNM$DCL_LOGICAL.)

The SHOW LOGICAL command performs iterative translations. If a logical name has more than one translation, then all translations at a level are displayed before going to the next level. Use the SHOW TRANSLATION command to display only the first translation found for a specified logical name.

The SHOW LOGICAL command executes an image and causes the current image (if any) to exit. Use the SHOW TRANSLATION command (which is built into the command interpreter) when you do not want to exit the current image.

QUALIFIERS

/ACCESS_MODE=mode

Displays names defined in the specified access mode and any inner access modes. You can specify one of the following keywords to indicate the access mode: USER_MODE, SUPERVISOR_MODE, EXECUTIVE_MODE, or KERNEL_MODE.

The default for this qualifier is USER_MODE; by default any definitions in all four access modes are displayed.

/ALL (default)

Indicates that all logical names in the specified logical name tables are to be displayed. If you do not enter the /PROCESS, /JOB, /GROUP, /SYSTEM, or /TABLE qualifier, all logical names in LNM$DCL_LOGICAL are displayed.

/DESCENDANTS
/NODESCENDANTS (default)

Controls whether the system displays names from the specified logical name table and any descendant tables. A descendant table is created by the CREATE/NAME_TABLE command, with the /PARENT_TABLE qualifier specifying its parent table. If you use the /DESCENDANTS qualifier, you must also use the /TABLE qualifier.

/FULL

Displays more detailed information for the specified logical name. The information includes the access mode, attributes, the translation, and the logical name table.

/GROUP

Indicates that only the group logical name table is to be searched. The /GROUP qualifier is synonymous with /TABLE=LNM$GROUP. If you specify the /GROUP qualifier and you do not also specify a logical name, all names in the group table are displayed.

/JOB

Indicates that only the job logical name table is to be searched. The /JOB qualifier is synonymous with /TABLE=LNM$JOB. If you specify the /JOB qualifier and you do not also specify a logical name, all names in the job logical name table are displayed.

/OUTPUT[=file-spec]
/NOOUTPUT

By default, the output of the SHOW LOGICAL command is sent to the current SYS$OUTPUT device (usually your terminal). To send the output to a file, use the /OUTPUT qualifier followed by a file specification.

The file specification may not include any wildcard characters. If you enter a partial file specification (for example, specifying only a directory), SHOW is the default file name and LIS is the default file type.

If you enter /NOOUTPUT, output is suppressed.

/PROCESS

Indicates that only the process logical name table is to be searched. The /PROCESS qualifier is synonymous with /TABLE=LNM$PROCESS. If you specify the /PROCESS qualifier and you do not also specify a logical name, all names in the process table are displayed.

/STRUCTURE
/NOSTRUCTURE (default)

Controls whether the system displays the "family tree" of all accessible logical name tables. The display includes the two logical name directory tables (process and system) and all logical name tables cataloged in these directory

tables. Any descendant logical name tables are shown under their parent tables.

If you specify /STRUCTURE, you cannot use any other qualifiers except /ACCESS_MODE, /FULL, and /OUTPUT.

/SYSTEM

Indicates that only the system logical name table is to be searched. The /SYSTEM qualifier is synonymous with /TABLE=LNM$SYSTEM. If you specify the /SYSTEM qualifier and you do not also specify a logical name, all names in the system table are displayed.

/TABLE=(name[,...])

Specifies the tables you want to search. If you specify only one table, you can omit the parentheses. Wildcards are allowed. Names with wildcards are used to match table names. Names without wildcards are treated both as table names and table search lists (whichever is appropriate).

You can use the /TABLE qualifier to specify the following:

- A user-defined logical name table (created with the CREATE/NAME_TABLE command).
- The process, group, or system logical name tables.
- The process or system directory tables.

If you specify the table name using a logical name that translates to more than one table, then each table is searched in the order specified. For example, if you specify SHOW LOGICAL/TABLE=LNM$FILE_DEV, and LNM$FILE_DEV is equated to LNM$PROCESS, LNM$JOB, LNM$GROUP, and LNM$SYSTEM, then the process, job, group, and system tables are searched, in that order.

If you do not specify the /TABLE qualifier, the default is /TABLE=LNM$DCL_LOGICAL.

EXAMPLES

1

```
$ SHOW LOGICAL/PROCESS
(LNM$PROCESS_TABLE)
  "SYS$COMMAND" = "_TTB4:"
  "SYS$DISK" = "WORK6:"
  "SYS$DISK" = "WORK6:"
  "SYS$ERROR" = "_TTB4:"
  "SYS$INPUT" = "_TTB4:"
  "SYS$LOGIN" = "WORK6:[ODONNELL]"
  "SYS$LOGIN_DEVICE" = "WORK6:"
  "SYS$OUTPUT" = "_TTB4:"
  "SYS$OUTPUT" = "DBA2:"
  "SYS$SCRATCH" = "WORK6:[ODONNELL]"
```

The SHOW LOGICAL command in this example displays all process logical names and their translations. (Note that /TABLE=LNM$PROCESS would produce the same display as /PROCESS.)

SHOW LOGICAL

```
2  $ SHOW LOGICAL INFILE
   "INFILE" = "WORK6:[LOGAN]PAYROLL.EXE" (LNM$PROCESS_TABLE)
```

The SHOW LOGICAL command in this example displays the translation for the logical name INFILE. The response indicates that the logical name was found in the process logical name table.

```
3  $ SHOW LOGICAL/GROUP
      .
      .
      .
```

The SHOW LOGICAL command in this example displays all group logical names and their translations. (Note that /TABLE=LNM$GROUP would produce the same display as /GROUP.)

```
4  $ SHOW LOGICAL/TABLE=SYSTEM  SYS$LIBRARY
   "SYS$LIBRARY" = "SYS$SYSROOT:[SYSLIB]" (LNM$SYSTEM_TABLE)
                 = "DOCD$:[SYSC.SYSLIB]"
```

The SHOW LOGICAL command in this example displays the translation of the logical name SYS$LIBRARY in the system table. The response indicates that SYS$LIBRARY is defined in the system table, and that the logical name has two translations.

```
5  $ SHOW LOGICAL/TABLE=LNM$GROUP/TABLE=LNM$SYSTEM SYS$DISK
   "SYS$DISK" = "ZZZ3:" (LNM$SYSTEM_TABLE)
```

The SHOW LOGICAL command in this example is qualified by both the /TABLE=LNM$GROUP and /TABLE=LNM$SYSTEM qualifiers. The response indicates that the logical name SYS$DISK was found in the system logical name table. When you enter two conflicting qualifiers, as in this example, only the last qualifier you specify is used.

```
6  $ SHOW LOGICAL/TABLE=LNM$PROCESS_DIRECTORY
```

The SHOW LOGICAL command in this example displays the logical names in the process directory table. Each name is either a table name, or a name that translates iteratively to a table.

SHOW MAGTAPE

Displays the current characteristics and status of a specified magnetic tape device.

FORMAT

SHOW MAGTAPE *device-name[:]*

PARAMETER

device-name[:]

Specifies the name of the magnetic tape device for which you want to display the characteristics and status.

QUALIFIER

/OUTPUT[=file-spec]
/NOOUTPUT

Specifies the file to which the display is written; by default, the display is written to the current SYS$OUTPUT device.

If you enter /OUTPUT with no file specification, or if you omit the file name or the file type, SHOW is the default file name and LIS the default file type. If you enter a file specification, you may not include any wildcard characters.

If you enter /NOOUTPUT, output is suppressed.

EXAMPLE

```
$ SHOW MAGTAPE MTA0:
  MTA0: UNKNOWN, DENSITY=800, FORMAT=Normal-11
        Odd Parity
```

The SHOW MAGTAPE command in this example displays the characteristics of the device MTA0:. The display shows the device type, density, and format (default or normal PDP–11).

It also displays the following characteristics:

Position lost	Write-locked
End-of-tape	Even parity
End-of-file	Odd parity
Beginning-of-tape	

SHOW MEMORY

Displays the availability and usage of those system resources that are related to memory.

FORMAT

SHOW MEMORY

PARAMETERS

None.

DESCRIPTION

The information provided by the SHOW MEMORY command can help you determine whether to change certain system memory resources to improve system performance. The system memory resources are as follows:

- Physical memory
- Process entry slots and balance slots
- Nonpaged and paged dynamic memory
- Space in paging and swap files

When the SHOW MEMORY command is executed, a display is written to SYS$OUTPUT. Depending on which qualifiers you specify, the display shows the following memory resource statistics:

- Physical memory usage
- Bad page list
- Number of pages allocated to VMS
- Slot usage
- Fixed-size pool areas (packets)
- Dynamic memory usage (bytes)
- Paging file usage (pages)
- Lookaside list
- Dynamic memory

QUALIFIERS

/ALL (default)

Displays all available information, that is, information displayed by the /FILES, /PHYSICAL_PAGES, /POOL, and /SLOTS qualifiers.

/FILES

Displays information about the use of each paging and swap file currently installed.

/FULL
When used with the /POOL or /FILES qualifier, displays additional information about the use of each pool area or paging and swap file currently installed. This qualifier is ignored unless the /FILES or /POOL qualifier is explicitly specified.

/OUTPUT[=file-spec]
/NOOUTPUT
Controls where the output of the command is sent. If you do not enter the qualifier, or if you enter /OUTPUT without a file specification, the output is sent to the current process default output stream or device, identified by the logical name SYS$OUTPUT.

If you enter /OUTPUT with a partial file specification (for example, specifying only a directory), SHOW is the default file name and LIS the default file type. If you enter a file specification, it may not include any wildcard characters.

If you enter /NOOUTPUT, output is suppressed.

/PHYSICAL_PAGES
Displays information about the amount of physical memory and the number of free and modified pages.

/POOL
Displays information about the usage of each dynamic memory (pool) area, including the amount of free space and the size of the largest contiguous block in each area.

/SLOTS
Displays information about the availability of PCB vector slots and balance slots.

EXAMPLES

1
```
$ SHOW MEMORY/PHYSICAL_PAGES

                System Memory Resources on  31-DEC-1988 16:11:30.76
Physical Memory Usage (pages):      Total❶      Free❷      In Use❸     Modified❹
Main Memory (32.00Mb)               65536      44233       20955          308

Of the physical pages in use, 10970 pages are permanently allocated to VMS.
```

Physical Memory Usage

Shows the use of physical memory.

❶Total	Displays the number of physical memory pages available for general system use. Multiport memory pages used for shared memory global sections, mailboxes, and common event blocks are not included in this number.
❷Free	Displays the number of pages on the free page list.
❸In Use	Displays the number of pages currently being used. This number is calculated by adding the number of pages on the free, modified, and bad lists and then subtracting that sum from the total number of available pages.
❹Modified	Displays the number of pages on the modified page list.

Bad Page List

Shows the contents of the bad page list. This display is written only when there are pages on the bad page list.

Total	Displays the number of pages on the bad page list.
Dynamic	Displays the number of memory errors detected after the system was booted.
I/O Errors	Displays the number of errors detected during page fault handling.
Static	Displays the number of memory errors detected during boot time scan.

By default, either single bit or double bit errors cause the pages to be removed during the boot time scan.

Pages Allocated to VMS

Any SHOW MEMORY display that includes the physical memory display concludes with the number of pages permanently allocated to the VMS operating system. These pages include nonpaged executive code and data, the PFN data base, nonpaged dynamic memory, the interrupt stack, and the system page table.

2
```
$ SHOW MEMORY/SLOTS

                System Memory Resources on 31-DEC-1988 16:11:35.31
Slot Usage (slots):                Total❶      Free❷  Resident❸    Swapped❹
Process Entry Slots                   75         28         46          1
Balance Set Slots                     70         26         44          0
```

Slot Usage (slots)

Displays the use of process entry slots and balance slots.

❶Total	Displays the number of process entry slots (the value of the SYSGEN parameter MAXPROCESSCNT) and balance slots (the value of the SYSGEN parameter BALSETCNT) permanently allocated when the system was bootstrapped.
❷Free	Displays the number of slots currently not in use.
❸Resident	Displays the number of slots currently used by memory-resident processes. The number of balance slots in use can never be any larger than the number of process entry slots in use because the SWAPPER and NULL processes have process entry slots but do not require balance slots.
❹Swapped	Displays the number of slots used by outswapped processes. For process entry slots, this number includes all processes that have been partially outswapped. For balance slots, this number includes those processes that have had their process bodies outswapped but have process headers that are still resident.

3

```
$ SHOW MEMORY/POOL

            System Memory Resources on  31-DEC-1988 16:11:39.97
Fixed-Size Pool Areas (packets):   Total❶      Free❷     In Use❸      Size❹
  Small Packet (SRP) List            624         42         582         96
  I/O Request Packet (IRP) List      500        257         243        160
  Large Packet (LRP) List             51         14          37        640

Dynamic Memory Usage (bytes):      Total❺      Free❻     In Use❼     Largest❽
  Nonpaged Dynamic Memory         161792       3488      158304       1936
  Paged Dynamic Memory             65536      29312       36224      29296
```

Fixed-Size Pool Areas (packets)

Shows the use of the nonpaged pool areas that consist of fixed-size packets, the so-called lookaside lists.

❶Total	Displays the total number of packets allocated for each list.
❷Free	Displays the number of available packets on each list.
❸In Use	Displays the number of packets in use on each list. This number is the total number of packets minus the number of free packets.
❹Size	Displays the fixed block size for each list.

Dynamic Memory Usage (bytes)

Shows the use of the nonpaged and paged pool areas that allocate variably sized blocks.

❺Total	Displays the total number of bytes set aside for each area.
❻Free	Displays the total amount of free space in each dynamic memory area.
❼In Use	Displays the amount of space currently allocated from each area. This number is simply the total size minus the number of free bytes.
❽Largest	Displays the size of the largest contiguous block in each area. For paged pool, this number represents the largest block that can be successfully allocated. For nonpaged pool, an allocation request larger than this number will cause nonpaged pool to grow (if other constraints allow growth).

4
```
$ SHOW MEMORY/FILES

            System Memory Resources on  14-APR-1988 16:11:45.83
Paging File Usage (pages):                      Free❷  Reservable❸      Total❹
  DISK$VMS02APR1:[SYS2.SYSEXE]SWAPFILE.SYS❶      256          256         4096
  DISK$VMS02APR1:[SYS2.SYSEXE]PAGEFILE.SYS      7613         6912         8192
```

Paging File Usage (pages)

Shows the usage of paging and swap files.

❶Name	Displays the complete file specification of each swap or paging file. The names of the primary paging file and the primary swapping file (if this file exists), and the files installed by the bootstrap operation are always displayed. The names of any secondary paging or swap files installed by the System Generation Utility (SYSGEN) are displayed only if the process using the SHOW MEMORY command has READ access to those files. If the process cannot read the file, the name is suppressed but the usage statistics are displayed.
❷Free	Displays the number of free blocks in each paging and swap file currently installed. Free blocks are blocks that may be physically allocated in the file.
❸Reservable	Displays the number of reservable blocks in each paging and swap file currently installed. Reservable blocks are blocks that may be logically claimed by a process for future physical allocation. A negative value indicates that the file may be overcommitted.
❹Total	Displays the total size of each paging and swap file.

When the /FULL qualifier is included on the SHOW MEMORY command for displays of pool areas or paging file usage, additional information is included in the pool or files displays.

5 $ SHOW MEMORY/POOL/FULL

```
                System Memory Resources on  31-DEC-1988 16:11:49.74
Small Packet (SRP) Lookaside List              Packets       Bytes       Pages
    Current Total Size❶                            624       59904         117
    Initial Size (SRPCOUNT)❷                        50        4800          10
    Maximum Size (SRPCOUNTV)❸                     1500      144000         282
    Free Space❹                                     46        4416
    Space in Use❺                                  578       55488
    Packet Size/Upper Bound (SRPSIZE)❻                          96
    Lower Bound on Allocation❼                                  48
I/O Request Packet (IRP) Lookaside List        Packets       Bytes       Pages
    Current Total Size❶                            500       80000         157
    Initial Size (IRPCOUNT)❷                       500       80000         157
    Maximum Size (IRPCOUNTV)❸                     1500      240000         469
    Free Space❹                                    259       41440
    Space in Use❺                                  241       38560
    Packet Size/Upper Bound (fixed)❻                            160
    Lower Bound on Allocation❼                                  112
Large Packet (LRP) Lookaside List              Packets       Bytes       Pages
    Current Total Size❶                             51       32640          64
    Initial Size (LRPCOUNT)❷                        37       23680          47
    Maximum Size (LRPCOUNTV)❸                       80       51200         100
    Free Space❹                                     14        8960
    Space in Use❺                                   37       23680
    Packet Size/Upper Bound (LRPSIZE + 64)❻                     640
    Lower Bound on Allocation❼                                  480
Nonpaged Dynamic Memory
    Current Size (bytes)❽          161792    Current Total Size (pages)❽       316
    Initial Size (NPAGEDYN)❾       149504    Initial Size (pages)❾             292
    Maximum Size (NPAGEVIR)❿       512000    Maximum Size (pages)❿            1000
    Free Space (bytes)⓫              4256    Space in Use (bytes)⓬          157536
    Size of Largest Block⓭           1936    Size of Smallest Block⓮            16
    Number of Free Blocks⓯             19    Free Blocks LEQU 32 Bytes⓰          6
Paged Dynamic Memory
    Current Size (PAGEDYN)❽         65536    Current Total Size (pages)❽       128
    Free Space (bytes)⓫             29312    Space in Use (bytes)⓬           36224
    Size of Largest Block⓭          29296    Size of Smallest Block⓮            16
    Number of Free Blocks⓯              2    Free Blocks LEQU 32 Bytes⓰          1
```

Lookaside List

Displays information about each nonpaged pool area that contains fixed-size blocks.

❶Current Size	Displays the current total size of each lookaside list. The "pages" column lists the number of physical pages permanently allocated to each list.
❷Initial Size	Displays the initial size of each lookaside list and the name of the SYSGEN parameter that controls this size.
❸Maximum Size	Displays the maximum size to which each lookaside list can grow and the name of the SYSGEN parameter that controls this size.
❹Free Space	Displays the amount of free space on each list.
❺Space in Use	Displays the amount of space currently allocated from each list.
❻Packet Size	Displays the fixed block size for each list and the relation between this size and a SYSGEN parameter. (Note that the block size for the I/O request packet lookaside list cannot be varied.) Allocation requests must be smaller than this size in order to be allocated from a given list.
❼Lower Bound	Displays the lower limit on allocation size permitted from each lookaside list.

Dynamic Memory

Shows the utilization of dynamic memory areas consisting of variably sized blocks.

❽Current Size	Displays the current size (in bytes and pages) of each dynamic memory area.
❾Initial Size	Displays the initial size of nonpaged dynamic memory and the name of the SYSGEN parameter (NPAGEDYN) that controls this size.
❿Maximum Size	Displays the maximum size to which nonpaged dynamic memory can grow and the name of the SYSGEN parameter (NPAGEVIR) that controls this size.
⓫Free Space	Displays the amount of free space in each dynamic memory area.
⓬Space in Use	Displays the amount of space currently allocated from each area.
⓭Largest Block	Displays the size of the largest contiguous area in each pool area.
⓮Smallest Block	Displays the size of the smallest hole (free block) in each pool area.
⓯Free Blocks	Displays the total number of free blocks in each pool area. The size of this number is a measure of pool fragmentation.
⓰Blocks LEQU 32	Displays the number of free blocks that are 32 bytes long or shorter. This number is another measure of pool fragmentation because while these small blocks are unlikely to be allocated, they contribute to the allocation time whenever an allocation request is made.

6
```
$ SHOW MEMORY/FILES/FULL

             System Memory Resources on  14-APR-1988 16:11:55.49
DISK$VMS02APR1:[SYS0.SYSEXE]SWAPFILE.SYS❶

    Free Blocks❷                256    Reservable Blocks❸              256
    Total Size (blocks)❹       4096    Paging File Number❺               1
    Swap Usage (processes)❻       0    Paging Usage (processes)❼         0
  This file is used exclusively for swapping.❽

DISK$VMS02APR1:[SYS0.SYSEXE]PAGEFILE.SYS❶
    Free Blocks❷               7611    Reservable Blocks❸             6912
    Total Size (blocks)❹       8192    Paging File Number❺               5
    Swap Usage (processes)❻       0    Paging Usage (processes)❼         0
  This file can be used for either paging or swapping.❽
```

Paging File Usage

Shows the usage of paging and swap files.

❶File Name	Displays the complete file specification of each paging or swap file, subject to the privilege restriction mentioned in the description of the normal display.
❷Free Blocks	Displays the number of free blocks in each paging and swap file currently installed. Free blocks are blocks that may be physically allocated in the file.
❸Reservable Blocks	Displays the number of reservable blocks in each paging and swap file currently installed. Reservable blocks are blocks that may be logically claimed by a process for future physical allocation. A negative value indicates that the file may be overcommitted.
❹Total Size	Displays the size of each paging and swap file.
❺File Number	Displays the internal paging file index assigned to each paging or swap file when it is installed.
❻Swap Usage	Displays the number of processes currently assigned space in this file for the purpose of swapping. Swap file assignment can change over the life of a process.
❼Paging Usage	Displays the number of processes currently paging to this file. Paging file assignment is made when a process is created and a process continues to page to that file. Note that the last two pieces of information are limited by the GROUP and WORLD privileges. That is, a process with neither privilege can only determine the paging and swap file assignments of itself and its subprocesses. A process with GROUP privilege can obtain this information about all processes that have the same group number. Only a process with WORLD privilege can obtain accurate paging and swap file information for the entire system.
❽File Usage	Displays a line of text that describes whether the file is used exclusively for swapping or is used for both paging and swapping.

```
7  $ SHOW MEMORY
```

The SHOW MEMORY command in this example displays all memory resource information, that is, information displayed by the /FILES, /PHYSICAL_MEMORY, /POOL, and /SLOTS qualifiers.

SHOW NETWORK

Displays the availability of the local node as a member of the network [1] and the addresses and names of all nodes that are currently accessible to the local node. The SHOW NETWORK command also displays link and cost relationships between the local node and other nodes in the network.

FORMAT

SHOW NETWORK

PARAMETERS

None.

DESCRIPTION

The SHOW NETWORK command displays the current network status. If the local node is a routing node, the following network characteristics are displayed: node, links, cost, hops, next hop to node, and, if the node is an area router, area and next hop to area. These characteristics are defined as follows:

Node	Identifies each available node in the network by its node address and node name.
Links	Shows the number of logical links between the local node and each available remote node.
Cost	Shows the total line cost of the path to a remote node. The cost for each line in the network is assigned by the system manager.
Hops	Shows the number of intermediate nodes plus the target node.
Next hop to node	Shows the outgoing physical line used to reach the remote node. The local node is identified by the term LOCAL. The node address and node name of the next hop to the target node are also displayed.
Area	Identifies each available area in the network by its area number. This characteristic is displayed only if the local node is an area router.
Next hop to area	Shows the outgoing physical line used to reach the remote area. This characteristic is displayed only if the local node is an area router. The local node is identified by the term LOCAL. The node address and node name of the next hop to the target area are also displayed.

If you enter the SHOW NETWORK command and the network is unavailable at that time, the following message is displayed:

```
Network unavailable
```

[1] DECnet–VAX is available under separate license.

SHOW NETWORK

QUALIFIER

/OUTPUT[=file-spec]
/NOOUTPUT

Controls where the output of the command is sent. If you do not enter the qualifier, or if you enter /OUTPUT without a file specification, the output is sent to the current process default output stream or device, identified by the logical name SYS$OUTPUT.

If you enter /OUTPUT with a partial file specification (for example, specifying only a directory), SHOW is the default file name and LIS the default file type. If you enter a file specification, it may not include any wildcard characters.

If you enter /NOOUTPUT, output is suppressed.

EXAMPLE

```
$ SHOW NETWORK

VAX/VMS Network Status for local node 2.161 ARAKIS on 15-APR-1988 09:18:03.07
The next hop to the nearest area router is node 2.62 ZEUS.
    Node        Links  Cost   Hops  Next Hop to Node
2.161  ARAKIS     0     0      0     Local   ->   2.161  ARAKIS
2.1    RAEL       0     8      1     UNA-0   ->   2.1    RAEL
2.2    PANGEA     0     8      1     UNA-0   ->   2.2    PANGEA
2.3    TWDEE      0    10      2     UNA-0   ->   2.63   AURORA
2.4    TWDUM      0     8      1     UNA-0   ->   2.4    TWDUM
2.11   NEONV      0     8      1     UNA-0   ->   2.11   NEONV
2.63   AURORA     0     8      1     UNA-0   ->   2.63   AURORA
             Total of 7 nodes.
```

If your local node is a nonrouting or end node and you enter the SHOW NETWORK command, the following message is displayed:

```
This is a nonrouting node, and does not have any network
information. The designated router for node _nodename is
node_number_name.
```

SHOW PRINTER

Displays the current settings for a printer.

FORMAT

SHOW PRINTER *device-name[:]*

PARAMETER

device-name[:]
Specifies the name of the printer for which settings are to be displayed.

DESCRIPTION

The SHOW PRINTER command displays the settings that are currently defined for the specified printer. Settings include the following:

- Printer type
- Page width
- Page length
- Line termination type
- Form feed setting
- Case setting
- Character translation settings
- Wrapping

Printer settings are established by the SET PRINTER command. You must have OPER (operator) privilege to use the SET PRINTER command.

QUALIFIER

/OUTPUT[=file-spec]
/NOOUTPUT
By default, the output of the SHOW PRINTER command is sent to the current SYS$OUTPUT device (usually your terminal). To send the output to a file, use the /OUTPUT qualifier followed by a file specification.

The file specification may not include any wildcard characters. If you enter a partial file specification (for example, specifying only a directory), SHOW is the default file name and LIS is the default file type.

If you enter /NOOUTPUT, output is suppressed.

EXAMPLE

```
$ SHOW PRINTER LPA0:
 Printer LPA0:, device type LP11, is online, allocated, spooled
 Error count                    0  Operations completed             880
 Owner process  "SYMBIONT_0001"  Owner UIC                       [0,0]
 Owner process ID      21C0008D  Dev Prot  S:RWLP,O:RWLP,G:RWLP,W:RWLP
 Reference count               2  Default buffer size               132
 Page width                  132  Page Length                       66
 No Carriage_return     Formfeed       Lowercase
 No Passall             No Wrap        Printall
 No Fallback
 Intermediate device: STAR$DBA1:
 Associated queue: LN01$PRINT
```

The SHOW PRINTER command in this example displays the settings for the printer LPA0.

SHOW PROCESS

Displays information about a process and any current subprocesses. If no qualifier is entered, only a basic subset of information is displayed: the time, process terminal, user name and UIC, process name and process identification, priority, default directory, and allocated devices.

Requires GROUP privilege to show other processes in the same group. Requires WORLD privilege to show processes outside your group.

FORMAT

SHOW PROCESS *[process-name]*

PARAMETER

process-name

Specifies the name of the process about which information is to be displayed. Process names can have from 1 to 15 alphanumeric characters. Process names are linked to group numbers. The specified process must have the same group number in its user identification code (UIC) as the current process.

You can also use the /IDENTIFICATION=PID qualifier to specify a process name. You cannot use the process name parameter and the /IDENTIFICATION qualifier together. If you do not include either the process name or the /IDENTIFICATION qualifier, the current process is assumed.

If you have GROUP or WORLD privilege, you can display information about processes other than your own. With GROUP privilege you can look at other processes in your group. With WORLD privilege you can look at processes outside of your group.

QUALIFIERS

/ACCOUNTING

Displays the accumulated accounting statistics for the current session.

/ALL

Displays the basic subset of information as well as accounting statistics, privileges, quotas, and subprocesses.

/CONTINUOUS

Displays continuously updated information about the process. While the continuous display is running, you can press the V key to display a map of the pages in the virtual address space of the process.

Each character displayed in the map represents the type of page. If the current program counter (PC) is in the page, the page type is indicated by an at (@) sign. Pages locked in the working set are indicated by the letter L. Global pages are indicated by the letter G. Other valid pages in the working set are indicated by an asterisk (*).

To terminate the continuous display, press the E key. To return to the original display, press the spacebar.

The /CONTINUOUS qualifier may not be used with the /OUTPUT qualifier.

/IDENTIFICATION=pid

Requires GROUP or WORLD privilege to access processes other than your own.

Displays information about the process with the specified PID (process identification). The PID is assigned by the system when the process is created. When you specify a PID, you can omit the leading zeros.

If you specify the /IDENTIFICATION qualifier, you cannot use the process name parameter. If, in addition, you specify either the /MEMORY or /SUBPROCESSES qualifiers, the process identification (PID) value must be that of the current process.

/MEMORY

Displays the process's use of dynamic memory areas. The /MEMORY qualifier is allowed only for the current process.

/OUTPUT[=file-spec]
/NOOUTPUT

By default, the output of the SHOW PROCESS command is sent to the current SYS$OUTPUT device (usually your terminal). To send the output to a file, use the /OUTPUT qualifier followed by a file specification.

The file specification may not include any wildcard characters. If you enter a partial file specification (for example, specifying only a directory), SHOW is the default file name and LIS is the default file type.

If you enter /NOOUTPUT, output is suppressed.

The /OUTPUT qualifier may not be used with the /CONTINUOUS qualifier.

/PRIVILEGES

Displays current privileges for the process.

/QUOTAS

Displays, for each resource, either a quota or a limit. The values displayed for quotas reflect any quota reductions resulting from subprocess creation. The values displayed for limits reflect the resources available to a process at creation.

/SUBPROCESSES

Displays the current subprocesses in hierarchical order. This qualifier can be used only for the current process.

EXAMPLES

1

```
$ SHOW PROCESS
 21-FEB-1988 15:35:19.39   VTA102:            User: MALIK
 Pid: 28200364   Proc. name: MALIK            UIC: [VMS,MALIK]
 Priority:   4   Default file spec: WORK5:[MALIK]
 Devices allocated: VTA102:
```

The SHOW PROCESS command in this example displays the basic subset of information for the current process. The information includes the following:

- Date and time the SHOW PROCESS command is entered
- Device name of the current SYS$INPUT device

- User name
- Process identification number (PID)
- Process name
- User identification code (UIC)
- Base execution priority
- Default device and directory
- Devices allocated to the process and volumes mounted, if any

2

```
$ SHOW PROCESS/ACCOUNTING
 21-FEB-1988 14:48:01.31   VTA102:            User: MALIK

Accounting information:
 Buffered I/O count:       4878  Peak working set size:        844
 Direct I/O count:         1284  Peak virtual size:           1176
 Page faults:              6100  Mounted volumes:                0
 Images activated:           22
 Elapsed CPU time:      0 00:01:20.51
 Connect time:          0 04:06:03.75
```

The SHOW PROCESS command in this example displays the accounting statistics for the current session.

3

```
$ SHOW PROCESS/PRIVILEGES
 21-FEB-1988 14:59:28.53   VTA102:            User: MALIK

Process privileges:
 GROUP                may affect other processes in same group
 TMPMBX               may create temporary mailbox
 NETMBX               may create network device

Process rights identifiers:
 INTERACTIVE
 DIALUP
```

The SHOW PROCESS command in this example displays the current privileges for the process.

4

```
$ SHOW PROCESS/QUOTAS
 21-FEB-1988 15:00:28.79   VTA102:            User: MALIK

Process Quotas:
Account name: VMS
 CPU limit:                      Infinite  Direct I/O limit:         6
 Buffered I/O byte count quota:     17904  Buffered I/O limit:       6
 Timer queue entry quota:              10  Open file quota:         31
 Paging file quota:                 24945  Subprocess quota:         8
 Default page fault cluster:           64  AST limit:               14
 Enqueue quota:                        30  Shared file limit:        9
 Max detached processes:               11  Max active jobs:         14
```

The SHOW PROCESS command in this example displays the available quotas and limits.

5 `$ SHOW PROCESS/SUBPROCESSES`

```
21-FEB-1988 15:44:59.39                    User: MALIK

Processes in this tree:

MALIK
  MALIK_1 (*)
    MALIK_2
```

The SHOW PROCESS command in this example shows the current process tree. The current process is indicated by the asterisk. Processes both below and above the current process are shown.

6 `$ SHOW PROCESS/CONTINUOUS FRED`

```
                          Process FRED                      12:26:53

   State                CUR               Working set            269
   Cur/base priority    8/4               Virtual pages         1713
   Current PC           7FFEE07E          Buffered I/O           646
   Current PSL          03C00000          Page faults           3417
   Current user SP      7FF785A4          Event flags       C8000007
                                                            C0000000
   PID                  226006C0          Direct I/O             246
   UIC                  [VMS,FRED]        CPU time       00:00:13.82
   SYS$SYSROOT:[SYSEXE]SHOW.EXE
```

In this example, the /CONTINUOUS qualifier causes the display of information about process FRED to be updated continuously.

7 `$ SHOW PROCESS/MEMORY`

```
21-FEB-1988 14:59:04.48   VTA102:          User: MALIK

Process Dynamic Memory Area
    Current Size (bytes)          25600   Current Total Size (pages)      50
    Free Space (bytes)            22698   Space in Use (bytes)          2902
    Size of Largest Block         22496   Size of Smallest Block          15
    Number of Free Blocks             7   Free Blocks LEQU 32 Bytes        3
```

The SHOW PROCESS command in this example displays the use of dynamic memory areas for the current process, MALIK. These areas are described as follows:

Current size	Displays the current size (in bytes and pages) of each dynamic memory area.
Free space	Displays the amount of free space in each dynamic memory area.
Space in use	Displays the amount of space currently allocated from each area.
Largest block	Displays the size of the largest contiguous area in each pool area.
Smallest block	Displays the size of the smallest hole (free block) in each pool area.
Free blocks	Displays the total number of free blocks in each pool area. The size of this number is a measure of pool fragmentation.
Blocks LEQU 32	Displays the number of free blocks that are 32 bytes long or shorter. This number is another measure of pool fragmentation. Although these small blocks are unlikely to be allocated, they contribute to the allocation time whenever an allocation request is made.

SHOW PROTECTION

Displays the current file protection to be applied to all new files created during the terminal session or batch job. You can change the default protection at any time with the SET PROTECTION command.

FORMAT SHOW PROTECTION

PARAMETERS *None.*

EXAMPLE

```
$ SHOW PROTECTION
  SYSTEM=RWED, OWNER=RWED, GROUP=RE, WORLD=NO ACCESS
$ SET PROTECTION=(GROUP:RWED,WORLD:RE)/DEFAULT
$ SHOW PROTECTION
  SYSTEM=RWED, OWNER=RWED, GROUP=RWED, WORLD=RE
```

The SHOW PROTECTION command in this example requests a display of the current protection defaults and the user identifiers; the SET PROTECTION /DEFAULT command changes the file access allowed to other users in the same group and to miscellaneous system users. The next SHOW PROTECTION command shows the modified protection defaults.

SHOW QUEUE

Displays information about queues and the jobs that are currently in queues.

Requires GROUP privilege to show all jobs in your group. Requires OPER privilege to show all jobs in all groups.

FORMAT

SHOW QUEUE *[queue-name]*

PARAMETER

queue-name

Specifies the name of the queue for which you want information displayed. Wildcard characters (* and %) are allowed. The default value for the queue-name parameter is the asterisk wildcard (*). If no queue name is specified, information on all queues is displayed.

DESCRIPTION

The SHOW QUEUE command displays the name, type, status, and attributes of batch and output queues as well as information about jobs in the queues. If you want status information about queues or a complete list of the jobs in those queues, use the SHOW QUEUE command. The default command is SHOW QUEUE.

A distinct advantage of the SHOW QUEUE command is that you can use different qualifier combinations to determine explicit information about queues. Using combinations that include the SHOW QUEUE qualifiers /BATCH, /BY_JOB_STATUS, /DEVICE, and /GENERIC allow you to extract this data. For example, the command SHOW QUEUE/GENERIC/BATCH displays information about all generic batch queues. For information about a specific job or to display only your jobs, use the SHOW ENTRY command.

When you enter the SHOW QUEUE command with no qualifiers, the system lists the names, types, and status of all the specified queues along with information about your jobs in those queues. In a cluster, all queues available for the cluster are listed.

QUALIFIERS

/ALL_JOBS

Displays all the jobs in the specified queues. If you do not specify a queue name, the /ALL_JOBS qualifier displays all job entries on all queues. To modify the display, combine this qualifier with the /BY_JOB_STATUS qualifier.

/BATCH

Displays all batch queues. Use the /BATCH qualifier in conjunction with other qualifiers to display specific information about particular batch queues.

/BRIEF

Displays a one line description of each queue and the jobs that are in it. This information includes the name, type, and status of each queue. It also includes the user name, entry number, and status for each job. The /FULL and /FILES qualifiers override /BRIEF.

/BY_JOB_STATUS=(keyword-list)

Displays jobs with the specified status. Specify the status with one or more of the following keywords:

EXECUTING	Requests the display of currently executing jobs.
HOLDING	Requests the display of jobs on hold. Holding status indicates that the job is being held in the queue indefinitely.
PENDING	Requests the display of jobs with pending status. Pending status indicates that the job is waiting its turn to execute.
RETAINED	Requests the display of jobs retained in the queue after execution. Retained status indicates that the job has completed, but it remains in the queue. For example, a job may be retained in the queue if there was an error during its execution.
TIMED_RELEASE	Requests the display of jobs on hold until a specified time. Timed release status indicates that the job is being held in the queue for execution at a future time.

Note that if you specify the qualifier as /BY, the system will only display queues that actually contain jobs.

/DEVICE[=(keyword-list)]

Displays a particular type of queue. Use the /DEVICE qualifier in conjunction with other qualifiers to display specific information about particular device queues.

Specify the type of device queue with one or more of the following keywords:

PRINTER	Requests the display of all print queues.
SERVER	Requests the display of all server queues.
TERMINAL	Requests the display of all terminal queues.

You can specify more than one keyword. If you do not specify a keyword, /DEVICE displays all printer, terminal, and server queues.

/FILES

Adds to the display the list of files associated with each job.

/FULL

Displays complete information about queues, the jobs contained in queues, and the files associated with the jobs. See /BRIEF.

/GENERIC

Displays all generic queues. A generic queue is not an execution queue. Its function is to hold jobs of a particular type (line printer jobs, for example) and direct them to execution queues for processing.

Use the /GENERIC qualifier in conjunction with other qualifiers to display specific information about particular generic queues. For example, use the /GENERIC qualifier along with the /BATCH qualifier to specify information about generic batch queues. Use the /GENERIC qualifier along with the /DEVICE qualifier to determine information concerning generic output queues.

/OUTPUT[=file-spec]
/NOOUTPUT

By default, the output of the SHOW QUEUE command is sent to the current SYS$OUTPUT device (usually your terminal). To send the output to a file, use the /OUTPUT qualifier followed by a file specification.

The file specification may not include any wildcard characters. If you enter a partial file specification (for example, specifying only a directory), SHOW is the default file name and LIS is the default file type.

If you enter /NOOUTPUT, output is suppressed.

/SUMMARY

Displays the total number of executing jobs, pending jobs, holding jobs, retained jobs, and timed release jobs for each queue. For output queues, the total block count for pending jobs is also shown.

EXAMPLES

1

```
$ SHOW QUEUE/ALL/BY_JOB_STATUS=executing
Printer queue KLEE$LCA0, on KLEE::KLEE$LCA0, mounted form DEFAULT

Printer queue SERIFA$LN01, on SERIFA::SERIFA$LPA0, mounted form
LN01$PORTRAIT (stock=DEFAULT)

  Jobname         Username     Entry  Blocks  Status
  -------         --------     -----  ------  ------
  APPLICA         DENNIS        2045     102  Printing

Batch queue SYS_TEX

  Jobname         Username     Entry  Status
  -------         --------     -----  ------
  CHAPTER1.JOB    MARTIN        1388  Executing
    .
    .
    .
```

The SHOW QUEUE command in this example displays all the queues on the system and any jobs within those queues that are currently executing. Notice that "printing" is an executing state for printer and terminal queues. "Processing" is an executing state for server queues.

2

```
$ SHOW QUEUE/FULL CAXTON_LPA0

Printer queue CAXTON_LPA0, on CAXTON::CAXTON_LPA0, mounted form
80_COLS (stock=BLUE)
    /BASE_PRIORITY=100
    /DEFAULT=(FEED,FLAG,FORM=40_COLS (stock=WHITE),TRAILER=ONE)
    /NOENABLE_GENERIC Lowercase /OWNER=[1,4] /PROTECTION=(S:E,O:D,G:R,W:W)

  Jobname          Username      Entry  Blocks  Status
  -------          --------      -----  ------  ------
  ACCOUNT          MARTIN          880      10  Printing
    Submitted  9-AUG-1988 12:49 /FORM=80_COLS (stock=BLUE)  /PRIORITY=100
  REPORT           MARTIN          858       4  Pending
    Submitted  8-AUG-1988 17:27 /PRIORITY=100
```

The SHOW QUEUE command in this example lists any current job entry you have on the printer queue CAXTON_LPA0. The /FULL qualifier lists the

submission information, the full file specification, and the current settings for both the job and the queue.

The job that is currently printing, ACCOUNT, is using a BLUE paper stock (as indicated by (stock=BLUE)). Notice that the default paper stock for the queue is WHITE (as indicated by (stock=WHITE)). Before the job ACCOUNT could be run, the operator had to stop the queue and mount the requested paper stock, BLUE. However, the next job on the queue, REPORT, expects the default paper stock of WHITE. Again, the operator must stop the queue and mount the correct paper stock. The job REPORT remains in a pending state until this is done.

3
```
$ SHOW QUEUE/SUMMARY/DEVICE=printer

Generic printer queue CLUSTER_PRINT
    1 holding,    1 timed release

Printer queue BREUER$LPB0, on BREUER::BREUER$LPB0, mounted form DEFAULT
    empty

Printer queue BAUHAU$LCA0, on BAUHAU::BAUHAU$LCA0, mounted form DEFAULT
    2 pending (8 blocks),    1 executing,    1 holding,    4 timed release
    1 retained

Generic printer queue LPS
    empty
    .
    .
    .
```

The SHOW QUEUE command in this example lists all the printer queues and summarizes the status of the jobs that are currently entered.

SHOW QUEUE/CHARACTERISTIC

Displays information about queue characteristics defined for the system. A characteristic is a user-defined attribute of a batch or output queue, such as ink color.

FORMAT

SHOW QUEUE/CHARACTERISTIC
[characteristic-name]

PARAMETER

characteristic-name

Specifies the name of a characteristic. Wildcard characters (* and %) are allowed. The default value for the characteristic-name parameter is the asterisk wildcard (*). Thus, information about all characteristics is displayed when you do not specify a characteristic name.

You create a characteristic name with the DEFINE/CHARACTERISTIC command. You can then assign it to a queue using the INITIALIZE/QUEUE, SET QUEUE, or START/QUEUE command with the /CHARACTERISTICS qualifier. You can also specify characteristics for a job using the /CHARACTERISTICS qualifier with the PRINT, SUBMIT, or SET ENTRY command.

DESCRIPTION

The SHOW QUEUE/CHARACTERISTIC command displays the characteristic name and number of characteristics defined for the system. To find out which characteristics are associated with a particular queue, use the SHOW QUEUE command with the /FULL qualifier.

QUALIFIERS

/OUTPUT[=filespec]
/NOOUTPUT

By default the output of the SHOW QUEUE/CHARACTERISTIC command is sent to the current SYS$OUTPUT device (usually your terminal). To send the output to a file, use the /OUTPUT qualifier followed by a file specification.

The file specification may not include any wildcard characters. If you enter a partial file specification (for example, specifying only a directory), SHOW is the default file name and LIS is the default file type.

If you enter /NOOUTPUT, output is suppressed.

EXAMPLES

1
```
$ SHOW QUEUE/CHARACTERISTIC
Characteristic name                Number
-------------------                ------
REDINK                                  0
COLOR_CHART                             1
TEXMAC_JOB                              2
BLUEINK                                 6
BROWNINK                               25
```

The SHOW QUEUE/CHARACTERISTIC command in this example displays all the characteristics that have been defined for this system.

2
```
$ SHOW QUEUE/CHARACTERISTIC *INK
Characteristic name                Number
-------------------                ------
REDINK                                  0
BLUEINK                                 6
BROWNINK                               25
```

The SHOW QUEUE/CHARACTERISTIC command in this example displays the name and number of all characteristics that end with INK.

SHOW QUEUE/FORM

Displays information about forms defined for the system. Forms define the size and type paper and the layout of text that are used for print jobs.

FORMAT

SHOW QUEUE/FORM *[form-name]*

PARAMETER

form-name

Specifies the name of the form. Wildcard characters are allowed. The default value for the form-name parameter is an asterisk (*) which means that the names of all forms on the system are displayed.

DESCRIPTION

The SHOW QUEUE/FORM command displays the name and number of forms defined for the system. To display the attributes associated with forms use the /FULL qualifier.

Use the SHOW QUEUE/FORM command to find out which forms are available on the system. To find out which forms are available on one or more queues, use the SHOW QUEUE command with the /FULL qualifier.

Form names are created by the DEFINE/FORM command. You can specify a particular form for a print job using the /FORM qualifier with the PRINT or SET ENTRY command. Forms are assigned to queues with the INITIALIZE /QUEUE, SET QUEUE, or START/QUEUE command.

QUALIFIERS

/BRIEF (default)

Displays a brief description (form names, numbers, and descriptions) about the forms on the system.

/FULL

Displays a full description (including paper size and margin settings) about the forms on the system.

/OUTPUT[=file-spec]
/NOOUTPUT

By default the output of the SHOW QUEUE/FORM command is sent to the current SYS$OUTPUT device (usually your terminal). To send the output to a file, use the /OUTPUT qualifier followed by a file specification.

The file specification may not include any wildcard characters. If you enter a partial file specification (for example, specifying only a directory), SHOW is the default file name and LIS is the default file type.

If you enter /NOOUTPUT, output is suppressed.

SHOW QUEUE/FORM

EXAMPLES

1

```
$ SHOW QUEUE/FORM DEFAULT
Form name                           Number   Description
---------                           ------   -----------
DEFAULT                                  0   System-defined default
```

The SHOW QUEUE/FORM command in this example displays only the default form.

2

```
$ SHOW QUEUE/FORM LN01*
Form name                           Number   Description
---------                           ------   -----------
LN01_LANDSCAPE (stock=DEFAULT)             105   132 by 66 (landscape)
LN01_LANDSCAPE_INDENTED (stock=DEFAULT)
                                           107   132 by 65 (landscape)
LN01_PORTRAIT (stock=DEFAULT)              106   80 by 60 (portrait)
```

The SHOW QUEUE/FORM command in this example displays the names of all forms, including the stock, that begin with LN01. The display includes the names, stock for each form, numbers, and brief descriptions of those forms.

3

```
$ SHOW QUEUE/FORM/FULL
Form name                           Number   Description
---------                           ------   -----------
132_51_STD (stock=DEFAULT)                 102   132 by 51 (standard short)
    /LENGTH=51 /MARGIN=(BOTTOM=6) /STOCK=DEFAULT /TRUNCATE /WIDTH=132
40_66_STD (stock=DEFAULT)                  103   40 by 66 (standard labels)
    /LENGTH=66 /MARGIN=(BOTTOM=6) /STOCK=DEFAULT /WIDTH=40
BLUE_PAPER_STOCK (stock=DIGITAL_8X11_STOCK1412TEA)
                                         22222   blue paper, DEC order# 22222
    /LENGTH=66 /MARGIN=(BOTTOM=6) /STOCK=DIGITAL_8X11_STOCK1412TEA
    /TRUNCATE  /WIDTH=80
DEFAULT                                    0   System-defined default
    /LENGTH=66 /MARGIN=(BOTTOM=6) /STOCK=DEFAULT /TRUNCATE /WIDTH=132
LN01_LANDSCAPE (stock=DEFAULT)             105   132 by 66 (landscape)
    /LENGTH=66 /STOCK=DEFAULT /WIDTH=132
LN01_LANDSCAPE_INDENTED (stock=DEFAULT)
                                           107   132 by 65 (landscape)
    /LENGTH=65 /SETUP=(LN01_TOP_MARGIN_150) /STOCK=DEFAULT /WIDTH=132
LN01_PORTRAIT (stock=DEFAULT)              106   80 by 60 (portrait)
    /LENGTH=60 /SETUP=(LN01_PORTRAIT) /STOCK=DEFAULT /WIDTH=80
MEMO (stock=DEFAULT)                       110   LN03 indented memo format
    /LENGTH=64 /MARGIN=(TOP=2,LEFT=5) /STOCK=DEFAULT /TRUNCATE /WIDTH=80
```

This SHOW QUEUE/FORM command also displays the names of all form types and stock for the system. By using the /FULL qualifier, you can see what image size has been set for each form type.

SHOW QUOTA

Displays the current disk quota that is authorized for a specific user on a specific disk. This display includes a calculation of the amount of space available and the amount of overdraft that is permitted.

Requires READ (R) access to the quota file in order to display the quotas of other users.

FORMAT

SHOW QUOTA

DESCRIPTION

The SHOW QUOTA command indicates whether a quota exists for any specific user on a specific disk. The display that results gives the quotas used, authorized, and available in blocks. The amount of overdraft permitted is also shown.

QUALIFIERS

/DISK[=device-name[:]]

Specifies the disk whose quotas are to be examined. By default, the current default disk (defined by SYS$DISK), is examined.

/USER=uic

Specifies which user's quotas are to be displayed. By default, the current user's quotas are displayed.

EXAMPLES

1

```
$ SHOW QUOTA
User [360,010] has 2780 blocks used, 7220 available,
of 10000 authorized and permitted overdraft of 500 blocks on DISK$
```

The SHOW QUOTA command in this example displays the amount of disk space authorized, used, and still available on the current default disk for the present user. The permitted overdraft in this example is 500 blocks.

2

```
$ SHOW QUOTA /USER=[360,007]/DISK=XXX1:
%SYSTEM-F-NODISKQUOTA, no disk quota entry for this UIC
```

The SHOW QUOTA command in this example displays the fact that the user with UIC [360,007] has no disk quota allocation on device XXX1.

3

```
$ SHOW QUOTA /USER=[360,111]
User [360,111] has 27305 blocks used, 2305 OVERDRAWN,
of 25000 authorized and permitted overdraft of 4000 blocks on DISK$
```

The SHOW QUOTA command in this example illustrates a user with an overdrawn quota.

SHOW RMS_DEFAULT

Displays the current default values for the multiblock count, the multibuffer count, the network transfer size, the prologue level, and the extend quantity.

FORMAT

SHOW RMS_DEFAULT

PARAMETERS

None.

DESCRIPTION

The SHOW RMS_DEFAULT command displays information that VMS RMS uses for file operations including the default values for the multibuffer count, the multiblock count, the network block count, the prologue level and the extend quantity. The command provides these values at both the current process level and at the system level.

The SHOW RMS_DEFAULT command lists the multibuffer count values for each of the file types, including the values for the three classes of sequential files (disk, magtape and unit record).

QUALIFIER

/OUTPUT[=file-spec]
/NOOUTPUT

Specifies the file to which the display is written (default is SYS$OUTPUT). Wildcard characters are not allowed. If you enter the /OUTPUT qualifier with a partial file specification (for example, specifying only a directory), SHOW is the default output file name and LIS the default output file type.

If you enter /NOOUTPUT, output is suppressed.

EXAMPLE

```
$ SHOW RMS_DEFAULT


          MULTI-  |                MULTIBUFFER COUNTS                | NETWORK
          BLOCK   | Indexed  Relative            Sequential          |  BLOCK
          COUNT   |                         Disk   Magtape  Unit Record |  COUNT
Process     0     |    0         0           0       0         0       |    0
System     16     |    0         0           0       0         0       |    8

           Prolog     Extend Quantity
Process      0              0
System       0              0
```

The SHOW RMS_DEFAULT command in this example shows a system multiblock count of 16 and a network block count of 8. These are typical values.

SHOW STATUS

The SHOW STATUS command in this example displays the current status of your process.

FORMAT

SHOW STATUS

PARAMETERS

None.

DESCRIPTION

Use the SHOW STATUS command to verify that your current process is running. The SHOW STATUS information can also indicate if the process is using an abnormal amount of CPU time, opening too many files, or accruing too many page faults.

The information displayed by SHOW STATUS is similar to that displayed by CTRL/T. (See the SET CONTROL command.)

EXAMPLE

```
$ SHOW STATUS
  Status on  15-APR-1988 12:56:48.68        Elapsed CPU :   0 00:00:55.02
  Buff. I/O :      5117    Cur. ws. :      300        Open files :         1
  Dir. I/O :        458    Phys. Mem. :    162        Page Faults :     8323
```

Displays the status of your process. The information includes the following:

- Current time and date
- Elapsed CPU time used by the current process
- Buffered I/O count
- Current working set size
- Open file count
- Direct I/O count
- Current amount of physical memory occupied
- Number of page faults

SHOW SYMBOL

Displays the value of the specified symbol.

FORMAT

SHOW SYMBOL *[symbol-name]*

PARAMETER

symbol-name

Specifies the name of the symbol whose value you want to display. You must specify a symbol name unless you use the /ALL qualifier. Wildcard characters are allowed in the symbol-name parameter.

DESCRIPTION

The SHOW SYMBOL command searches for the specified symbol name in the following places. First, it looks in the local symbol table of the current command level. Next, it searches the local symbol tables of preceding command levels. Last, it searches the global symbol table. It displays the first match it finds.

The /LOCAL and /GLOBAL qualifiers override the search order.

QUALIFIERS

/ALL

Displays the current values of all symbols in the specified symbol table (/LOCAL or /GLOBAL). If you specify /ALL and do not specify either /LOCAL or /GLOBAL, the SHOW SYMBOL command displays the contents of the local symbol table for the current command level.

/GLOBAL

Searches only the global symbol table for the specified symbol name. If you specify both the /ALL and /GLOBAL qualifiers, all names in the global symbol table are displayed.

/LOCAL

Searches only the local symbol table for the current command level for the specified symbol name. If you specify both the /ALL and /LOCAL qualifiers, all names in the local symbol table for the current command level are displayed.

/LOG (default)
/NOLOG

Controls whether the system generates an informational message if the symbol value has been truncated. The value is truncated if it exceeds 255 characters.

EXAMPLES

1
```
$ SHOW SYMBOL PURGE
  PURGE = "PURGE/KEEP=2"
```

The SHOW SYMBOL command in this example displays the current value of the symbol name PURGE. The command interpreter first searches the local symbol table for the current command level, then local symbol tables for preceding command levels, and finally the global symbol table. The single equal sign following PURGE means it is a local symbol.

2
```
$ SHOW SYMBOL/GLOBAL/ALL
  TIME == "SHOW TIME"
  LOG == "@LOG"
  $RESTART == "FALSE"
  $SEVERITY == "1"
  $STATUS == "%X00000001"
```

The SHOW SYMBOL command in this example displays all the symbols defined in the global symbol table. Note that the symbols $RESTART, $STATUS, and $SEVERITY, which are maintained by the system, are also displayed.

3
```
$ SHOW SYMBOL/LOCAL TIME
%DCL-W-UNDSYM, undefined symbol
```

The SHOW SYMBOL command in this example searches only the local symbol table for the symbol TIME. The response indicates that TIME currently has no value.

SHOW SYSTEM

Displays status information about current processes: the time, process name and identification, processing state, priority, total process I/O, cumulative processor time used, cumulative page faults, amount of physical memory being used, and type of process.

FORMAT

SHOW SYSTEM

PARAMETERS

None.

DESCRIPTION

The SHOW SYSTEM command displays information about processes on the system. It also checks to see if the machine is a multiprocessor. A machine is considered a multiprocessor if the following conditions are true:

- The SYSGEN parameter MULTIPROCESSING is not equal to zero
- The machine accommodates more than one CPU

If the machine is a multiprocessor, the SHOW SYSTEM command includes information about the multiprocessing environment. Each process in the currently executing state (indicated by the CUR symbol) reveals which processor it is running on. It does this by including a number beside the CUR symbol.

QUALIFIERS

/BATCH

Displays all batch jobs in the system.

/FULL

Displays the user identification code (UIC) in addition to the default information. The UIC is displayed underneath the process name.

/NETWORK

Displays all network processes in the system.

/OUTPUT[=file-spec]
/NOOUTPUT

By default, the output of the SHOW SYSTEM command is sent to the current SYS$OUTPUT device (usually your terminal). To send the output to a file, use the /OUTPUT qualifier followed by a file specification.

The file specification may not include any wildcard characters. If you enter a partial file specification (for example, specifying only a directory), SHOW is the default file name and LIS is the default file type.

If you enter /NOOUTPUT, output is suppressed.

/PROCESS (default)
Displays all processes in the system.

/SUBPROCESS
Displays all subprocesses in the system.

EXAMPLE

```
$ SHOW SYSTEM
VMS V5.0  on node CAXTON 31-DEC-1988 15:10:31.02   Uptime    0 12:06:30
  Pid     Process Name    State  Pri      I/O       CPU        Page flts Ph.Mem
22200081 SWAPPER         HIB     16        0    0 00:01:07.92         0    0
22200202 Meg             LEF      4     6206    0 00:03:52.53     17509  174
22200085 ERRFMT          HIB      8     4123    0 00:00:45.35       145  191
22200086 CLUSTER_SERVER  HIB     10       23    0 00:00:01.63       130  167
22200087 OPCOM           RWAST    9     1114    0 00:00:39.60       189  274
22200088 JOB_CONTROL     HIB      8     6662    0 00:02:29.09       368  474
22200089 CONFIGURE       HIB     13       84    0 00:00:00.64       148  235
2220008A VAXsim_Monitor  HIB      8     1543    0 00:00:20.55       312  137
2220008C PARTITION_CHKR  LEF      4     3833    0 00:00:14.27       132  153
2220008D SYMBIONT_0001   HIB      4     2866    0 00:16:26.65      6751  329
2220008E Cerb Servant    LEF      4    12412    0 00:03:08.40       854  165
2220008F Monitor         LEF     15     1151    0 00:00:28.64      2396  350
22200090 NETACP          HIB      9    14820    0 00:02:42.85      8464  350
22200091 EVL             HIB      5      209    0 00:00:07.90      9996   49   N
22200092 REMACP          HIB      9      194    0 00:00:01.18       123   59
22200113 BIGELOW         PFW      4    16505    0 00:04:17.12     23318  912
22200119 OHARE           LEF      9     7680    0 00:01:36.09      8689  233
2220011D BRATZ           LEF      9     6276    0 00:00:43.66      2006  200
2220019E PENN            LEF      4     2646    0 00:01:59.11      4847  280
222001A4 TAMMY           LEF      9    13514    0 00:01:49.52      3011  512
222001A6 C_EMACS         HIB      7    16165    0 00:05:10.86      6026  639   S
222001A7 PERKINS         LEF      4     7572    0 00:02:13.84     16487  195
22200128 SAPP            LEF      4    15634    0 00:07:34.16     39591  450
2220022C ZIPPY           LEF      7     2297    0 00:00:52.86      6668  300
2220012E GENIUS          LEF      9      671    0 00:00:12.02      1197  200
2220022F BORSE           LEF      4      741    0 00:00:13.13      2394  150
22200231 MSCPmount       LEF      6     1211    0 00:00:48.40      3793   92
222000B2 MCGOY           LEF      4    11710    0 00:01:54.90     12798  180
222000B5 pf cobbs ttg2   LEF      4     6076    0 00:01:26.10      8418  220
22200236 COBBS           HIB      8    39982    0 00:00:31.73      5419  452
22200236 COBBS_1         CUR 1    4      251    0 00:00:05.80       560  330
22200237 MAIL_6188       LEF      6      236    0 00:00:04.58       609  269   N
22200238 MAIL_6189       LEF      4      236    0 00:00:04.58       542  261   N
22200239 Mike Smith TTG  COM      4      220    0 00:00:05.42       844  154
222000BC Greg A-z        LEF      9     9873    0 01:35:16.42     16557  949
222001C5 BUNNYRABBIT     LEF      9      492    0 00:00:12.00      2171  150
222000C8 CARP            LEF      8     9381    0 00:05:44.12     22090  227
2220014A CATFISH         LEF      9     9512    0 00:02:03.49     10264  227
222000CB TUNA            LEF      9     5676    0 00:01:56.31      9468  526
222000CC HANGNAIL        LEF      9    43511    0 00:16:30.55     55396  163
222000D2 BAZOO           LEF      4    17315    0 00:05:25.43     62535  300
222000DE KENNY           LEF      7     4887    0 00:01:39.36      8800  544
2220015F MACY            CUR 0    4    13671    0 00:03:10.51     19027  302
22200162 _TTH1:          LEF      4    19344    0 00:20:57.28     54894  269
222000E4 BATCH_970       LEF      6      669    0 00:00:09.79       639  386   B
22200167 EPPY            HIB      5     3399    0 00:01:01.07     10761  192
222000E9 Greg A_z        LEF      6    12899    0 00:01:52.69      6624  447
```

SHOW SYSTEM

The SHOW SYSTEM command in this example displays all processes on the system.

The information includes the following:

- Process identification code (PID)—a 32-bit binary value that uniquely identifies a process.
- Process name—a 1-15 character string used to identify a process.
- Process state—the activity level of the process, such as COM (computing), HIB (hibernation), LEF (local event flag) wait, or CUR (if the process is current). If a multiprocessing environment exists, the display shows the CPU ID of the processor on which any current process is executing.
- Current priority—the priority level assigned to the process (the higher the number, the higher the priority). [1]
- Total process I/O count [1]—the number of I/O operations involved in executing the process. This consists of both the direct I/O count and the buffered I/O count.
- Charged CPU time [1]—the amount of CPU time that a process has used thus far.
- Number of page faults [1]—the number of exceptions generated by references to pages which are not in the process's working set.
- Physical memory occupied [1]—the amount of space in physical memory that the process is currently occupying.
- Process indicator—letter B indicates a batch job; letter S indicates a subprocess; letter N indicates a network process.
- User identification code (UIC)—an eight-digit octal number assigned to a process. This is only displayed if the /FULL qualifier is specified.

[1] This information is displayed only if the process is currently in the balance set; if the process is not in the balance set, these columns contain the following message:

- swapped out -

SHOW TERMINAL

Displays the current characteristics of a specific terminal. Each characteristic corresponds to an option of the SET TERMINAL command.

FORMAT

SHOW TERMINAL *[device-name[:]]*

PARAMETER

device-name[:]

Specifies the name of the terminal for which you want the characteristics displayed. The default is your terminal (SYS$COMMAND).

DESCRIPTION

The SHOW TERMINAL command displays the information about terminal settings for such things as terminal speed, width, number of lines, graphics, and device type.

Note that, SHOW TERMINAL does not describe terminal fallback characteristics if any are activated. If the Terminal Fallback Facility (TFF) is enabled, you can invoke the Terminal Fallback Utility and issue the subcommand SHOW TERMINAL/FALLBACK. See the *VMS Terminal Fallback Utility Manual* for more information.

QUALIFIERS

/OUTPUT[=file-spec]
/NOOUTPUT

Controls where the output of the command is sent. If you do not enter the qualifier, or if you enter /OUTPUT without a file specification, the output is sent to the current process default output stream or device, identified by the logical name SYS$OUTPUT.

If you enter /OUTPUT with a partial file specification (for example, specifying only a directory), SHOW is the default file name and LIS the default file type. If you enter a file specification, it may not include any wildcard characters.

If you enter /NOOUTPUT, output is suppressed.

/PERMANENT

Requires LOG_IO or PHY_IO privilege.

Displays the permanent characteristics of the terminal.

SHOW TERMINAL

EXAMPLE

```
$ SHOW TERMINAL
Terminal: _TTE4:       Device_Type: VT102         Owner: FRANKLIN
Physical Terminal: _LTA49
   Input:   9600      LFfill:  0      Width:  80      Parity: None
   Output:  9600      CRfill:  0      Page:   24
Terminal Characteristics:
   Interactive        Echo               Type_ahead         No Escape
   No Hostsync        TTsync             Lowercase          Tab
   Wrap               Scope              No Remote          Eightbit
   Broadcast          No Readsync        No Form            Fulldup
   No Modem           No Local_echo      No Autobaud        Hangup
   No Brdcstmbx       No DMA             No Altypeahd       Set_speed
   Line Editing       Overstrike editing No Fallback        No Dialup
   No Secure server   No Disconnect      No Pasthru         No Syspassword
   No SIXEL Graphics  Soft Characters    Printer port       Numeric Keypad
   ANSI_CRT           No Regis           No Block_mode      Advanced_video
   Edit_mode          DEC_CRT            DEC_CRT2          No DEC_CRT3
```

In this example, the SHOW TERMINAL command displays the characteristics of this specific terminal. If you are displaying statistics about a terminal allocated to another user, the input, output, LFfill, CRfill, width, page, and parity statistics are not shown.

SHOW TIME

Displays the current date and time. The DAY element is optional.

FORMAT

SHOW [DAY]TIME

PARAMETERS

None.

EXAMPLE

```
$ SHOW TIME
  4-FEB-1988 00:03:45
```

The SHOW TIME command in this example displays the current date and time.

SHOW TRANSLATION

Displays the first translation found for the specified logical name. You can specify the tables that are searched.

Requires READ (R) access to a logical name table to display information about any logical name cataloged in that table.

FORMAT

SHOW TRANSLATION *logical-name*

PARAMETER

logical-name

Specifies the logical name whose translation you want to display.

DESCRIPTION

The SHOW TRANSLATION command searches one or more logical name tables for a specified logical name and returns the equivalence name of the first match found. You can specify the tables you want to search. If you do not specify a table, SHOW TRANSLATION searches the tables defined by LNM$DCL_LOGICAL.

LNM$DCL_LOGICAL contains the list of logical name tables and the order in which they are searched. Unless LNM$DCL_LOGICAL has been redefined for your process, the process, job, group, and system logical name tables are searched, in that order. The first match found is displayed. (To see how LNM$DCL_LOGICAL is defined for your process, type the command SHOW LOGICAL /TABLE=LNM$DIRECTORIES LNM$DCL_LOGICAL.)

If a table contains more than one entry with the same name, but each name has a different mode, then the translation for the name with the outermost (least privileged) mode is returned.

The SHOW TRANSLATION command is similar to the SHOW LOGICAL command. However, the SHOW TRANSLATION command is executed within the DCL command interpreter (the SHOW LOGICAL command calls an image). Therefore, the SHOW TRANSLATION command does not cause the current image to exit and does not deassign user mode logical names. Also, the SHOW TRANSLATION command does not display iterative translations of a name.

QUALIFIERS

/TABLE=name

Searches the specified table. The default is /TABLE=LNM$DCL_LOGICAL.

If you specify the table name using a logical name that translates to more than one table, then each table is searched in the order specified until a match is found.

EXAMPLES

1

```
$ SHOW TRANSLATION PAYROLL
  PAYROLL = DISK1:[ACCOUNTS.WORKING]FACTOR1.DAT;37 (LNM$PROCESS_TABLE)
```

The SHOW TRANSLATION command in this example displays the translation for the logical name PAYROLL and also displays the name of the table where the logical name was found. In this example, PAYROLL was found in LNM$PROCESS_TABLE, the process logical name table.

2

```
$ DEFINE DISK  DBA1:
$ DEFINE/GROUP DISK  DBA2:
$ SHOW TRANSLATION DISK
  DISK = DBA1:(LNM$PROCESS_TABLE)
```

The DEFINE commands in this example place entries for the logical name DISK in both the process and group logical name tables. Then, the SHOW TRANSLATION command shows the translation associated with the logical name DISK. By default, the process, job, group, and system tables are searched (in that order). The first match found is displayed. The logical name DISK from the process table (LNM$PROCESS_TABLE) is displayed because it is found before the name DISK in the group table.

3

```
$ RUN ORION
CTRL/Y
$ SHOW TRANSLATION TERMINAL
  TERMINAL = _TTT3: (LNM$PROCESS_TABLE)
$ CONTINUE
```

The RUN command in this example executes the image ORION.EXE. After CTRL/Y interrupts the image, the SHOW TRANSLATION command displays a logical name assignment. The CONTINUE command resumes the execution of the image.

4

```
$ SHOW TRANSLATION/TABLE=LNM$SYSTEM USER
  USER = "DBA2:"  (LNM$SYSTEM_TABLE)
```

The SHOW TRANSLATION command in this example displays the translation for the logical name USER. Because a table name is specified, the SHOW TRANSLATION command does not use the default search order. Only the specified table, LNM$SYSTEM, is searched. LNM$SYSTEM is the system logical name table.

5

```
$ DEFINE/TABLE=LNM$PROCESS_DIRECTORY MYPROC -
_$ TEST_TABLE, LNM$PROCESS
$ SHOW TRANSLATION/TABLE=MYPROC FILER
  FILER = "[SMITH.FILER]"   (TEST_TABLE)
```

In this example, MYPROC defines a list of logical name tables that you want searched. It asks the system to first search TEST_TABLE (a user-defined table) and then to search LNM$PROCESS (the process logical name table). MYPROC is stored in LNM$PROCESS_DIRECTORY, the process directory table. When you enter the SHOW TRANSLATION command to find FILER in the MYPROC table, the tables TEST_TABLE and LNM$PROCESS are searched, in that order. The first match found is displayed.

SHOW USERS

Displays the terminal name, user name, and process identification code (PID) of interactive users on the system.

FORMAT

SHOW USERS *[username]*

PARAMETER

username

Specifies the user about whom you want information. If you specify a string, all users whose user names begin with the string are displayed. For example, if you specify the string MAR, all user names that begin with MAR are displayed. If no user exists whose name matches the specified string, an informational message to that effect is displayed.

If you omit the username parameter, a list of all interactive users is displayed.

QUALIFIER

/OUTPUT[=file-spec]
/NOOUTPUT

By default, the output of the SHOW USERS command is sent to the current SYS$OUTPUT device (usually your terminal). To send the output to a file, use the /OUTPUT qualifier followed by a file specification.

The file specification may not include any wildcard characters. If you enter a partial file specification (for example, specifying only a directory), SHOW is the default file name and LIS is the default file type.

If you enter /NOOUTPUT, output is suppressed.

EXAMPLES

1

```
$ SHOW USERS
          VAX/VMS Interactive Users
            31-DEC-1988 12:48:51.14
    Total number of interactive users = 14

  PID     Username     Process Name    Terminal
202000B3  <LOGIN>      _VTA9:          VTA9:          TTA7:
204000C4  AHO          AHO             VTA21:         LTA8:
2040013A  ETZEL        M Etzel VTA43:  VTA43:         TTA1:
20400138  FRISSELLE    FRISSELLE       VTA42:         TTA5:
20400095  HOBBS        cw hobbs        VTA1:          TTD5:
204000DC  HUANG        _VTA32:         VTA32:         LTA11:
204000B9  KUEHN        KUEHN           VTA14:         TTB0:
20400123  MALIK        MALIK           VTA38:         TTB6:
20400113  MCLAUGHLIN   MCLAUGHLIN      VTA35:         TTB3:
204000BC  MURRAY       MURRAY          VTA16:         TTA2:
204000C2  OPERATOR     OPERATOR        VTA20:         OPA0:
2040012C  PERRON       PERRON          VTA40:         TTA4:
2040013D  POLLACK      POLLACK         VTA45:         TTA3:
20400097  STEEVES      STEEVES         VTA2:          LTA1:
```

The SHOW USERS command in this example displays the process

identification code (PID), the user name, process name, and terminal names (both virtual and physical) of all interactive users currently on the system. A user name of <LOGIN> indicates that someone is in the process of logging in.

2
```
$ SHOW USERS GOSHGARIAN
          VAX/VMS Interactive Users
           31-DEC-1988 16:45:14.14
    Total number of interactive users = 32

  PID     Username     Process Name    Terminal
20200115 GOSHGARIAN    GOSHGARIAN       VTA3:          TTA7:
```

The SHOW USERS command in this example displays the process identification code (PID), the user name, process name, and terminal names of the interactive user GARGARIAN.

3
```
$ SHOW USERS J
          VAX/VMS Interactive Users
           31-DEC-1988 16:45:11.66
    Total number of interactive users = 32

  PID     Username     Process Name    Terminal
202000DB JANE         JANE            VTA19:         TTC1:
2020011D JUDY         JUDY            VTA20:         TTA6:
```

The SHOW USERS command in this example displays the process identification code (PID), the user name, process name, and terminal names of all interactive users whose user names begin with the letter J.

SHOW WORKING_SET

Displays the working set limit, quota, and extent assigned to the current process.

FORMAT

SHOW WORKING_SET

PARAMETERS

None.

QUALIFIER

/OUTPUT[=file-spec]
/NOOUTPUT

Controls where the output of the command is sent. If you do not enter the qualifier, or if you enter /OUTPUT without a file specification, the output is sent to the current process default output stream or device, identified by the logical name SYS$OUTPUT.

If you enter /OUTPUT with a partial file specification (for example, specifying only a directory), SHOW is the default file name and LIS the default file type. If you enter a file specification, it may not include any wildcard characters.

If you enter /NOOUTPUT, output is suppressed.

EXAMPLE

```
$ SHOW WORKING_SET
  Working Set      /Limit= 180   /Quota= 350           /Extent= 1200
  Adjustment enabled    Authorized Quota= 350  Authorized Extent= 1200
```

In this example, the response to the SHOW WORKING_SET command indicates that the current process has a working set limit of 180 pages, a quota of 350 pages and that the current quota is equal to the authorized limit (350 pages). It also shows that the current process has a working set extent of 1200 and that the current extent is equal to the authorized limit (1200).

SORT

Invokes the Sort/Merge Utility (SORT) to reorder the records in a file into a defined sequence and to create either a new file of the reordered records or an address file by which the reordered records can be accessed. For a complete description of the Sort/Merge Utility, including more information about the SORT command, see the *VMS Sort/Merge Utility Manual*.

FORMAT

SORT *input-file-spec[,...] output-file-spec*

SPAWN

Creates a subprocess of the current process. Portions of the current process context are copied to the subprocess.

The RESOURCE_WAIT state is required to spawn a process. Requires TMPMBX or PRMMBX user privilege. The SPAWN command does not manage terminal characteristics. The SPAWN and ATTACH commands cannot be used if your terminal has an associated mailbox.

FORMAT

SPAWN *[command-string]*

PARAMETER

command-string

Specifies a command string of less than 132 characters that is to be executed in the context of the created subprocess. When the command completes execution, the subprocess terminates and control returns to the parent process. If both a command string and the /INPUT qualifier are specified, the specified command string executes before additional commands are obtained from the /INPUT qualifier.

DESCRIPTION

The SPAWN command creates a subprocess of your current process with the following attributes copied from the parent process:

- All symbols except $RESTART, $SEVERITY, and $STATUS
- Key definitions
- The current keypad state
- The current prompt string
- All process logical names and logical name tables except those explicitly marked CONFINE or those created in executive or kernel mode
- Default disk and directory
- Current SET MESSAGE settings
- Current process privileges
- Control and verification states

Note that some attributes, such as the process's current command tables, are *not* copied.

When the subprocess is created, the process-permanent open files and any image or procedure context are *not* copied from the parent process. The subprocess is set to command level 0 (DCL level with the current prompt).

If you do not specify the /PROCESS qualifier, the name of this subprocess is composed of the same base name as the parent process and a unique number. For example, if the parent process name is SMITH, the subprocess name can be SMITH_1, SMITH_2, and so on.

The LOGIN.COM file of the parent process is not executed for the subprocess because the context is copied separately, allowing quicker initialization of the subprocess. When the /WAIT qualifier is in effect, the parent process remains in hibernation until the subprocess terminates or returns control to the parent by way of the ATTACH command.

More than one process simultaneously attempts to use the same input or output stream when several processes share that stream and you terminate a subprocess to which you are not currently attached; or when you terminate a process that is not spawned from the process to which you are currently attached.

You should use the LOGOUT command to terminate the subprocess and return to the parent process. You can also use the ATTACH command (see ATTACH) to transfer control of the terminal to another process in the subprocess tree, including the parent process. (The SHOW PROCESS /SUBPROCESSES command displays the processes in the subprocess tree and points to the current process.)

Note: Because a tree of subprocesses can be established using the SPAWN command, you must be careful when terminating any process in the tree. When a process is terminated, all subprocesses below that point in the tree are automatically terminated.

Qualifiers used with the SPAWN command must directly follow the command verb. The command string parameter begins after the last qualifier and continues to the end of the command line.

QUALIFIERS

/CARRIAGE_CONTROL
/NOCARRIAGE_CONTROL

Determines whether carriage return/line feed characters are prefixed to the subprocess's prompt string. By default, SPAWN copies the current setting of the parent process.

/CLI=cli-file-spec
/NOCLI

Specifies the name of a command language interpreter (CLI) to be used by the subprocess. The default CLI is the same as the parent process (defined in SYSUAF). If you specify /CLI, the attributes of the parent process are copied to the subprocess.

The CLI you specify must be located in SYS$SYSTEM and have the file type EXE.

/INPUT=file-spec

Specifies an input file containing one or more DCL commands to be executed by the spawned subprocess. File type defaults to COM and no wildcards are allowed in the file specification. Once processing of the input file is complete, the subprocess is terminated. If both a command string and the /INPUT qualifier are specified, the specified command string executes before additional commands are obtained from the /INPUT qualifier. If neither is specified, SYS$INPUT is assumed (in which case a SPAWN/NOWAIT command is aborted if CTRL/Y is typed to abort something running in your parent process).

You cannot explicitly specify non-record-oriented process permanent files (NRO PPFs) with the /INPUT qualifier. The system displays an error message when it encounters such a file as the value for /INPUT.

Note that when NRO PPFs are used as implicit input (that is, /INPUT is not specified and SYS$INPUT is a NRO PPF), the SPAWN command can succeed. The following chart shows what happens.

Process Type	SYS$INPUT	Implicit Input
Interactive	NRO PPF	SYS$COMMAND
Noninteractive	NRO PPF	Null Device
Any	Any other	SYS$INPUT

If SYS$INPUT is a terminal, it cannot have an associated terminal mailbox.

/KEYPAD (default)
/NOKEYPAD

Copies keypad key definitions and the current keypad state from the parent process. By default, if you have established key definitions or states with the DEFINE/KEY or SET KEY commands, these settings are copied to the subprocess. Use the /NOKEYPAD qualifier if you do not want the key settings to be copied.

/LOG (default)
/NOLOG

Displays the assigned subprocess name and any messages indicating transfer of control between processes.

/LOGICAL_NAMES (default)
/NOLOGICAL_NAMES

Copies process logical names and logical name tables to the subprocess. By default, all process logical names and logical name tables are copied to the subprocess except those explicitly marked CONFINE or created in executive or kernel mode.

/NOTIFY
/NONOTIFY (default)

Controls whether a message is broadcast to your terminal notifying you that your subprocess has completed or aborted. This qualifier should not be used unless you specify the /NOWAIT qualifier. /NOTIFY cannot be specified when the SPAWN command is executed from within a noninteractive process.

Note that messages broadcast as a result of using the /NOTIFY qualifier are considered to be DCL messages. Therefore, if SET BROADCAST=NODCL is in effect, all such notification messages are suppressed.

/OUTPUT=file-spec

Specifies the output file to which the results of the SPAWN operation are written. No wildcards can be used in the file specification. (Do not specify SYS$COMMAND as a file specification for /OUTPUT when using the /NOWAIT qualifier; both parent and subprocess output will be displayed simultaneously on your terminal.)

You cannot explicitly specify non-record-oriented process permanent files (NRO PPFs) with the /OUTPUT qualifier. The system displays an error message when it encounters such a file as the value for /OUTPUT.

Note that when NRO PPFs are used as implicit output, the SPAWN command can succeed. The following chart shows what happens.

Process Type	SYS$OUTPUT	Implicit Output
Any	NRO PPF	Mailbox transmitting records for parent to write to its current SYS$OUTPUT device
Any	Any other	SYS$OUTPUT

If you omit the /OUTPUT qualifier, output is written to the current SYS$OUTPUT device.

/PROCESS=subprocess-name

Specifies the name of the subprocess to be created. If you omit the /PROCESS qualifier, a unique process name is assigned with the same base name as the parent process and a unique number. The default subprocess name format is: username_n. If you specify a process name that already exists, an error message is displayed. If the /LOG qualifier has been specified, the assigned name of the subprocess is displayed.

/PROMPT[=string]

Specifies the prompt string for DCL to use in the subprocess. The default is the prompt of the parent process.

The string can consist of more than one character. All valid ASCII characters can be used in the string. The string must be enclosed in quotation marks if it contains spaces, special characters, or lowercase characters. Otherwise, letters are automatically converted to uppercase, and leading and trailing spaces are removed.

If no string is specified, the DCL default prompt string "$ " is used for the subprocess.

/SYMBOLS (default)
/NOSYMBOLS

Determines whether global and local symbols (except $RESTART, $SEVERITY, and $STATUS) are passed to the subprocess. $RESTART, $SEVERITY, and $STATUS symbols are *never* passed to the subprocess.

/TABLE=command-table

Specifies the name of an alternate command table to be used by the subprocess.

/WAIT (default)
/NOWAIT

Requires that you wait for the subprocess to terminate before you enter another DCL command. The /NOWAIT qualifier allows you to enter new commands while the subprocess is running. (Use the /OUTPUT qualifier with the /NOWAIT qualifier to avoid displaying both parent and subprocess output on the terminal simultaneously.)

Note that specifying the /NOWAIT qualifier causes both input and output to be shared with the parent process. If the input device is a terminal, control characters, such as CTRL/T or CTRL/Y, also affect all subprocesses sharing the input device. CTRL/Y, for example, interrupts all such subprocesses.

This problem may be avoided by specifying /INPUT=NL:.

EXAMPLE

```
$ RUN MYPROG
      .
      .
      .
$ CTRL/Y
$ SPAWN MAIL
%DCL-S-SPAWNED, process SMITH_1 spawned
%DCL-S-ATTACHED, terminal now attached to process SMITH_1
MAIL> READ
   .
   .
   .
MAIL> EXIT
%DCL-S-RETURNED, control returned to process SMITH
$ CONTINUE
```

The SPAWN command in this example allows you to enter the VMS Mail Utility without terminating the currently running program. After you exit from MAIL, control is returned to the parent process.

START/CPU

Starts the specified secondary processor or processors in a VMS multiprocessing system. The /CPU qualifier is required.

Applies only to VMS multiprocessing systems. Requires change mode to kernel (CMKRNL) privilege.

FORMAT

START/CPU *[cpu-id,...]*

PARAMETER

cpu-id

Decimal value representing the identity of a processor in a VMS multiprocessing system. In a VAX 8300 system, for instance, the CPU ID is the VAXBI node number of the processor; in a VAX 8800, the CPU ID of the left processor is 1 and that of the right processor is 0. If you do not specify a CPU ID and do not include the /ALL qualifier, the START/CPU command selects a single available processor to join the multiprocessing system.

DESCRIPTION

The START/CPU command starts a secondary processor in a VMS multiprocessing system.

You can issue a START/CPU command only for processors in the STOPPED or TIMOUT state, as represented by the SHOW CPU command. Otherwise, the START/CPU command has no effect.

QUALIFIER

/ALL

Selects all remaining processors in the system's available set to join the multiprocessing system.

EXAMPLES

1 `$ START/CPU`

The START/CPU command in this example selects a single inactive processor from the set of those processors that are currently available but inactive. When it completes its initialization, the selected processor becomes part of the system's active set and is capable of scheduling and executing processes.

2 `$ START/CPU 4,7`

The START/CPU command in this example selects the processors with CPU IDs 4 and 7, if they are currently available and inactive. When they complete initialization, these processors become part of the system's active set and are capable of scheduling and executing processes.

```
3  $ START/CPU/ALL
```

The START/CPU/ALL command in this example selects all remaining inactive and available processors. When they complete initialization, these processors become part of the system's active set and are capable of scheduling and executing processes.

START/QUEUE

Starts or restarts the specified queue after it has been initialized. The /QUEUE qualifier is required.

Requires OPER privilege or EXECUTE (E) access to the specified queue.

FORMAT

START/QUEUE *queue-name[:]*

PARAMETER

queue-name[:]
Specifies the name of the queue to be started or restarted.

DESCRIPTION

The START/QUEUE command restarts a queue that has been stopped for some reason. To use the START/QUEUE command, the queue must already have been initialized with the INITIALIZE/QUEUE command.

In general, to start a newly created queue, use the INITIALIZE/QUEUE /START command to both initialize and start the queue with one command. If you want to start the queue at a later time, you can use the INITIALIZE /QUEUE command to initialize the queue and subsequently enter the START /QUEUE command to start the queue.

You can also use the START/QUEUE command to restart a queue after it has been stopped. When you enter the START/QUEUE command, you can reset the queue features.

Note that any qualifier that can be used with the INITIALIZE/QUEUE command can be specified with the START/QUEUE command, except for /START. The START/QUEUE command can override options that have been specified with the INITIALIZE/QUEUE command, such as /ENABLE_GENERIC, /DEFAULT, and /FORM_MOUNTED.

If the specified queue is running when you enter the START/QUEUE command, the system returns an error message. Use the SET QUEUE command to change the attributes of a running queue.

When you specify more than one of the following qualifiers, the system processes them in the following order:

- /TOP_OF_FILE
- /BACKWARD or /FORWARD
- /SEARCH
- /ALIGN

QUALIFIERS

/ALIGN[=(option[,...])]

Prints alignment pages that enable the operator to align properly the forms in the printer or terminal. Use this qualifier to restart an output queue from a paused state. Possible options are as follows:

MASK	Displays alphabetic characters as x's and numbers as 9's; nonalphanumeric characters are not masked. The default is not to mask.
n	Specifies the number of alignment pages to print. The value of *n* can be from 1 to 20; the default is 1.

/BACKWARD=n

Restarts a print queue *n* pages before the current page; *n* defaults to 1. Use this qualifier in restarting an output queue from a paused state.

/BASE_PRIORITY=n

Specifies the base process priority at which jobs are initiated from a batch queue or the base priority of the symbiont process for printer, terminal, or server queues. By default, if you omit the qualifier, jobs are initiated at the same priority as the base priority established by DEFPRI at system generation (usually 4). The value n can be any decimal value from 0 through 15.

/BATCH
/NOBATCH (default)

Indicates that this is a batch queue. The /NOBATCH qualifier cancels the effect of a previous /BATCH qualifier on the same command. It is supported in this release for compatibility with VMS Version 4.n.

The function of the /[NO]BATCH qualifier has been incorporated into the /[NO]BATCH qualifier of the INITIALIZE/QUEUE command. DIGITAL recommends that you use this command to determine queue type and that existing command procedures using START/QUEUE/[NO]BATCH be updated.

/BLOCK_LIMIT=([lowlim,]uplim)
/NOBLOCK_LIMIT

Restricts the size of print jobs that can be executed on a printer or terminal queue. You must specify at least one of the parameters.

The lower parameter is a decimal number referring to the minimum number of blocks that are accepted by the queue for a print job. The upper parameter is a decimal number referring to the maximum number of blocks that are accepted by the queue for a print job. If a job contains fewer blocks than the number specified by the lower parameter or more blocks than the number specified by the upper parameter, the job remains pending until the block limit for the queue is changed, enabling the job to execute.

If you specify only an upper limit for jobs, you can omit the parentheses. For example, /BLOCK_LIMIT=1000 means that only jobs with 1000 blocks or less will execute in the queue. To specify only a lower job limit, you must use two consecutive quotation marks to indicate the upper specifier. For example, /BLOCK_LIMIT=(500,"") means any job with 500 or more blocks execute in the queue. You can specify both a lower and upper limit. For example, /BLOCK_LIMIT=(200,2000) means that jobs with less than 200 blocks or more than 2000 blocks do not run in the queue.

The /NOBLOCK_LIMIT qualifier cancels the /BLOCK_LIMIT setting previously established for that queue.

/CHARACTERISTICS=(characteristic[,...])
/NOCHARACTERISTICS

Specifies one or more characteristics for processing jobs on the queue. Each time you specify /CHARACTERISTIC, all previously set characteristics are erased. A queue must have all the characteristics specified for the job or the job remains pending. Only the characteristics specified with the qualifier are now established for the queue. If only one characteristic is specified, you can omit the parentheses.

Queue characteristics are installation-specific. The characteristic parameter can be either a value from 0 through 127 or a characteristic name that has been defined by the DEFINE/CHARACTERISTIC command.

When users include the /CHARACTERISTICS qualifier with a PRINT or SUBMIT command, all the characteristics they specify must also be specified for the queue executing the job. If not, the job remains pending in the queue until the queue characteristics are changed or they delete the entry with the DELETE/ENTRY command. Users need not specify every characteristic of a queue with a PRINT or SUBMIT command as long as the ones they specify are a subset of the characteristics set for that queue. The job also runs if no characteristics are specified.

/CLOSE

Prevents jobs from being entered in the queue through PRINT or SUBMIT commands or as a result of requeue operations. To allow jobs to be entered, use the /OPEN qualifier. Whether a queue accepts or rejects new job entries is independent of the queue's state (such as paused, stopped, stalled). When a queue is marked closed, jobs executing continue to execute. Jobs already pending in the queue continue to be candidates for execution.

/CPUDEFAULT=time

Indicates the default CPU time limit for batch jobs. Time can be specified as delta time, 0, NONE, or INFINITE. You can specify up to 497 days of delta time. Both the value 0 and the keyword INFINITE allow unlimited CPU time (subject to the restrictions imposed by the /CPUMAXIMUM qualifier or the user authorization file); the keyword NONE indicates that no time limit is needed. The value for time cannot exceed the CPU time limit set by the /CPUMAXIMUM qualifier.

See Section 1.4 of the *VMS DCL Concepts Manual* for information on specifying delta times.

/CPUMAXIMUM=time

Indicates the maximum CPU time limit for batch jobs. The /CPUMAXIMUM qualifier overrides the time limit specified in the user authorization file (UAF). Time can be specified as delta time, 0, NONE, or INFINITE. You can specify up to 497 days of delta time. Both the value 0 and the keyword INFINITE allow unlimited CPU time; the keyword NONE specifies that no time limit is needed. See Section 1.4 of the *VMS DCL Concepts Manual* for information on specifying delta times.

A CPU time limit for processes is specified by each user record in the system UAF. You can also specify the following: a default CPU time limit for all jobs in a given queue and a maximum CPU time limit for all jobs in a given queue. See Table DCL–1 for information on what action is taken for each value specified and for the possible combinations of specifications.

/DEFAULT=(option[,...])
/NODEFAULT

Establishes defaults for certain options of the PRINT command. Defaults are specified by the list of options. If you specify only one option, you can omit the parentheses. Once an option is set for the queue by the /DEFAULT qualifier, users do not have to specify that option in their PRINT commands. The /DEFAULT qualifier can not be used with the /GENERIC qualifier. Possible options are as follows:

[NO]BURST[=keyword]	Specifies whether to print burst pages (flag pages printed over the paper's perforations for easy identification of individual files in a print job). The keyword ALL (the default) places burst pages before each printed file in the job. The keyword ONE places a burst page before the first printed file in the job.
[NO]FEED	Specifies whether a form-feed is automatically inserted at the end of a page
[NO]FLAG[=keyword]	Specifies whether to print flag pages (containing the job entry number, the name of the user submitting the job, and so on). The keyword ALL places flag pages before each printed file in the job. The keyword ONE places a flag page before the first printed file in the job.
FORM=type	Specifies the default form for a printer, terminal, or server queue. If a job is not submitted with an explicit form definition, then this form is used to process the job. The systemwide default form, form=0, is the default value for this keyword. See also /FORM_MOUNTED.
[NO]TRAILER[=keyword]	Specifies whether to print trailer pages. The keyword ALL places trailer pages after each printed file in the job. The keyword ONE places a trailer page after the last printed file in the job.

/DESCRIPTION=string
/NODESCRIPTION (default)

A string of up to 255 characters used to provide operator-supplied information about the queue.

If the string contains alphanumeric, underscore, or dollar sign characters it must be enclosed in quotation marks (").

The /NODESCRIPTION qualifier removes any descriptive text that may have been associated with the queue.

/DISABLE_SWAPPING
/NODISABLE_SWAPPING (default)

Controls whether batch jobs executed from a queue can be swapped in and out of memory.

/ENABLE_GENERIC (default)
/NOENABLE_GENERIC

Allows files queued to a generic queue that does not specify explicit queue names in the /GENERIC qualifier to be placed in this execution queue for processing.

/FORM_MOUNTED=type

Specifies the form type for a printer, terminal, or server queue. If the stock of the mounted form is not identical to the stock of the default form, as indicated by the DCL command qualifier /DEFAULT=FORM=type, then all jobs submitted to this queue without an explicit form definition enter a pending state. If a job is submitted with an explicit form and the stock of the explicit form is not identical to the stock of the mounted form, then the job enters a pending state. In both cases, the pending state is maintained until the stock of the mounted form of the queue is identical to the stock of the form associated with the job.

Specify the form type using a numeric value or a form name that has been defined by the DEFINE/FORM command. Form types are installation-specific. The /FORM_MOUNTED qualifier can not be used with the /GENERIC qualifier.

/FORWARD=n

Advances the specified number of pages before resuming printing the current file in the current job; the default is 1. Use this qualifier to restart an output queue from a paused state.

/GENERIC[=(queue-name[,...])]
/NOGENERIC (default)

Specifies that this is a generic queue and that jobs placed in it can be moved for processing to compatible execution queues. The /GENERIC qualifier optionally accepts a list of target execution queues that have been previously defined. For a generic batch queue, these target queues must be batch execution queues. For a generic output queue, these target queues must be output execution queues, but can be of any type (printer, server, or terminal). For example, a generic printer queue can feed a mixture of printer and terminal execution queues.

If you do not specify any target queues with the /GENERIC qualifier, jobs can be moved to any execution queue that (1) is initialized with the /ENABLE_GENERIC qualifier, and (2) is the same type (batch, printer, server, or terminal) as the generic queue. Moreover, for a generic server queue, an additional check is made: the symbiont named with the /PROCESSOR qualifier must be the same for both the generic and execution queues.

The /GENERIC qualifier is used in conjunction with either the /BATCH or /DEVICE qualifiers to define the queue as a generic batch, printer, server, or terminal queue. If neither /BATCH nor /DEVICE is specified on creation of a generic queue, it becomes a generic printer queue by default.

/JOB_LIMIT=n

Specifies the number of batch jobs that can be executed concurrently from the queue. The job limit default value for n is 1.

/LIBRARY=file-name
/NOLIBRARY

Specifies the file name for the device control library. When you are initializing a symbiont queue, you can use the /LIBRARY qualifier to specify an alternate device control library. The default library is SYS$LIBRARY:SYSDEVCTL.TLB. You can specify only a file name as the parameter of the /LIBRARY qualifier. The system always assumes that the location of the file is in SYS$LIBRARY and that the file type is TLB.

/NEXT

Restarts the queue with the next job. By default, the job that was executing when the queue stopped resumes printing if it has not been deleted. Use the /NEXT qualifier to abort the current job and start with the next job in the queue.

/ON=[node::]device[:] (printer, terminal, server queue)
/ON=node:: (batch queue)

Specifies the node or device, or both, on which this execution queue is located. For batch queues, only the node name can be specified. You can include both the node name and the device name for printer and terminal queues. By default, a queue executes on the same node from which you first start the queue. The default device parameter is the same as the queue name.

The node name is used only in VAXcluster systems; it must match the node name specified by the SYSGEN parameter SCSNODE for the processor on which the queue executes.

/OPEN (default)

Allows jobs to be entered in the queue through PRINT or SUBMIT commands or as the result of requeue operations. To prevent jobs from being entered, use the /CLOSE qualifier. Whether a queue accepts or rejects new job entries is independent of the queue's state (such as paused, stopped, stalled).

/OWNER_UIC=uic

Requires OPER privilege.

Enables you to change the UIC of the queue. Specify the UIC using standard UIC format as described in Section 8.1 of the *VMS DCL Concepts Manual*. The default UIC is [1,4].

/PROCESSOR=file-name
/NOPROCESSOR

Allows users to specify their own print symbionts. The file name specifier can be any valid file name. The system supplies the device and directory name SYS$SYSTEM as well as the file type EXE. If you use this qualifier for an output queue, it specifies that the symbiont image to be executed is SYS$SYSTEM:file-name.EXE. By default, SYS$SYSTEM:PRTSMB.EXE is executed.

If you use this qualifier for a generic queue, it specifies that the generic queue can place jobs only on queues established as server queues and that are executing the specified symbiont image.

The /NOPROCESSOR qualifier cancels the effect of a previous /PROCESSOR setting.

/PROTECTION=(codes)

Requires OPER privilege.

Specifies the protection of the queue. By default, the queue protection is (SYSTEM:E, OWNER:D, GROUP:R, WORLD:W). If you include only one protection code, you can omit the parentheses.

/RECORD_BLOCKING (default)
/NORECORD_BLOCKING

Determines whether the symbiont can concatenate (or block together) output records for transmission to the output device. If you specify /NORECORD_BLOCKING, the symbiont is directed to send each formatted record in a separate I/O request to the output device. For the standard VMS print symbiont, record blocking can have a significant performance advantage over single-record mode.

/RETAIN[=option]
/NORETAIN (default)

Retains jobs in the queue in a completed status after they have executed. Possible options are as follows:

ALL	Retains all jobs in the queue after execution (default)
ERROR	Retains in the queue only jobs that complete unsuccessfully

The /NORETAIN qualifier enables you to reset the queue to the default.

/SCHEDULE=[NO]SIZE

Specifies whether pending jobs in a printer or terminal queue are scheduled for printing based on the size of the job. When the default, /SCHEDULE=SIZE, is in effect, shorter jobs are printed before longer ones.

If you enter this command while there are pending jobs in any queue, its effect on future jobs is unpredictable.

/SEARCH="search-string"

Resumes printing the current file of the current job on the first page containing the specified string. The string can be from 1 through 63 characters and must be enclosed in quotation marks. Use this qualifier in restarting an output queue from a paused state.

/SEPARATE=(option[,...])
/NOSEPARATE

Specifies the job separation defaults for a printer or terminal queue. The /SEPARATE qualifier cannot be used with the /GENERIC qualifier. The job separation options are as follows:

[NO]BURST	Specifies whether a burst page prints at the beginning of every job. Specifying BURST also results in a flag page being printed.
[NO]FLAG	Specifies whether a flag page prints at the beginning of every job.
[NO]TRAILER	Specifies whether a trailer page prints at the end of every job.
[NO]RESET=(module[,...])	Specifies a job reset sequence for the queue. The specified modules from the device control library are used to reset the device each time a job reset occurs.

/TERMINAL
/NOTERMINAL

Indicates that the output queue is a terminal queue. The /NOTERMINAL qualifier cancels the effect of a previous /TERMINAL qualifier on the same command. It is supported in this release for compatibility with VMS Version 4.n and may be retired in the future.

The function of the /[NO]TERMINAL qualifier has been incorporated into the /[NO]DEVICE qualifier of the INITIALIZE/QUEUE command. DIGITAL recommends that you use this command to determine queue type and that existing command procedures using START/QUEUE/[NO]TERMINAL be updated.

/TOP_OF_FILE

Resumes printing at the beginning of the file that was current when the queue paused. Use this qualifier only when restarting an output queue from a paused state.

/WSDEFAULT=n

Defines a working set default for a batch job. The value set by this qualifier overrides the value defined in the user authorization file (UAF) of any user submitting a job to the queue.

Possible values are a positive integer in the range 1 through 65,535, 0, or the word NONE can be specified for n. If you specify 0 or NONE, the working set default value becomes the value specified either in the UAF or by the SUBMIT command (if specified). For more information see Table DCL–2.

You can also specify this qualifier for an output queue. Used in this context, it establishes the working set default of the symbiont process for a printer, terminal, or server queue when the symbiont process is created.

/WSEXTENT=n

Defines a working set extent for the batch job. The value set by this qualifier overrides the value defined in the user authorization file (UAF) of any user submitting a job to the queue.

Possible values are a positive integer in the range 1 through 65,535, 0, a positive integer in the range 1 through 65,535, 0, or the word NONE can be specified for n. If you specify 0 or NONE, the working set value becomes the value specified either in the UAF or by the SUBMIT command (if specified). For more information see Table DCL–2.

You can also specify this qualifier for an output queue. Used in this context, it establishes the working set extent of the symbiont process for a printer, terminal, or server queue when the symbiont process is created.

/WSQUOTA=n

Defines the working set page size (working set quota) for a batch job. The value set by this qualifier overrides the value defined in the user authorization file (UAF) of any user submitting a job to the queue.

Possible values are: a positive integer in the range 1 through 65,535, 0, or the word NONE as the value for n. If 0 or NONE is specified for n, the working set quota defaults to the value specified either in the UAF or by the SUBMIT command (if specified). A working set default size and a working set quota (maximum size) are included in each user record in the system UAF and can be specified for individual jobs and/or for all jobs in a given queue. For more information see Table DCL–2.

You can also specify this qualifier for an output queue. Used in this context, it establishes the working set quota of the symbiont process for a printer, terminal, or server queue when the symbiont process is created.

EXAMPLES

1
```
$ START/QUEUE/BATCH SYS$BATCH
```

The START/QUEUE command in this example starts the batch queue named SYS$BATCH. The /BATCH qualifier indicates that this is a batch queue.

2
```
$ STOP/QUEUE  LPA0
$ START/QUEUE/TOP_OF_FILE  LPA0
```

The STOP/QUEUE command in this example suspends operation of the printer queue LPA0. Then the START/QUEUE/TOP_OF_FILE command resumes operation. The file that was being printed when the queue was stopped is started again from the beginning.

3
```
$ INITIALIZE/QUEUE  LPA0
      .
      .
      .
$ START/QUEUE/DEFAULT=FLAG  LPA0
```

The INITIALIZE/QUEUE command in this example initializes the queue named LPA0. Later, the START/QUEUE command starts the queue. The /DEFAULT qualifier requests that a flag page precede each file in each job.

4
```
$ START/QUEUE/DEFAULT=FORM=LN01_PORTRAIT LN01_PRINT
```

The START/QUEUE command in this example restarts the the LN01_PRINT queue with the default form LN01_PORTRAIT.

START/QUEUE/MANAGER

Starts the queue manager for the batch/print facility and opens the job queue manager file. After the system is bootstrapped, you must execute this command before you can execute any other queue management or job submission command. The /QUEUE qualifier is optional, but you must specify the /MANAGER qualifier.

For more information, see the *Guide to Maintaining a VMS System.*

Requires both OPER and SYSNAM privileges.

FORMAT

START/QUEUE/MANAGER *[file-spec]*

PARAMETER

file-spec

Specifies the name of the file to contain information about batch and print jobs, queues, and form definitions. The file specification parameter is used in VAXcluster systems or for specifying an alternate system job queue file. The default file specification is SYS$SYSTEM:JBCSYSQUE.DAT. Any elements that you omit from the file specification default to those of SYS$SYSTEM:JBCSYSQUE.DAT. No wildcard characters are permitted in the file specification.

DESCRIPTION

The START/QUEUE/MANAGER command is generally included in the system startup procedure. You can omit the file specification if you want your system to use the default job queue manager file.

You must also ensure that the definition of each execution queue (by the INITIALIZE/QUEUE or START/QUEUE command) contains the /ON qualifier. The node name specified with the /ON qualifier must match the system's node name as defined by the SYSGEN parameter SCSNODE.

In a VAXcluster, you must include the file specification parameter to cause each system to access the same job queue manager file on a shared disk volume. The file specification must include at least the device and directory names.

QUALIFIER

/BUFFER_COUNT=n

Specifies the number of buffers in a local buffer cache to allocate for performing I/O operations to the system job queue file. Specify a positive integer in the range of 1 through 127, or 0. If 0 is specified, the default value of 50 is used.

/EXTEND_QUANTITY=n

Specifies the number of blocks by which the system job queue file is extended, when necessary. This value is also used as the initial allocation size when the queue file is created. Specify a positive integer in the range of 10 through 65,535, or 0. If 0 is specified, the default value of 100 is used.

/NEW_VERSION
/NONEW_VERSION (default)

Specifies that a new version of the job queue manager file be created to supersede an existing version. All jobs in the previous version are lost if a new version is specified. The new file contains no information until you enter a subsequent INITIALIZE/QUEUE command.

/RESTART
/NORESTART (default)

The /RESTART qualifier specifies that the queue manager be restarted automatically on recovery from a job controller abort. In addition, batch and output queues are restored to the states that existed prior to the interruption of service. The job queue manager file that is opened is the same file that was open before the abort. Upon restarting, the job controller uses the default values for the /EXTEND_QUANTITY and /BUFFER_COUNT qualifiers. Previously set values are lost.

When the job controller incurs an internal fatal error, the process aborts and restarts itself. By default, the queue manager is not restarted. Intervention by a user with OPERATOR privilege is necessary to restart the queue manager and to restore the queueing environment using START/QUEUE/MANAGER and appropriate START/QUEUE commands.

Note that in order to prevent a looping condition, the job controller does not restart the queue manager if it detects an error within two minutes of starting the queue manager.

EXAMPLES

1 `$ START/QUEUE/MANAGER`

The START/QUEUE/MANAGER command in this example opens the default job queue manager file.

2 `$ START/QUEUE/MANAGER DUA5:[SYSQUE]`

The START/QUEUE/MANAGER command in this example opens the job queue manager file JBCSYSQUE.DAT on the cluster-accessible disk volume DUA5, in directory SYSQUE. You must mount the disk before you enter the START/QUEUE/MANAGER command.

STOP

Terminates execution of a command, an image, a command procedure, a command procedure that was interrupted by CTRL/Y, or a detached process or subprocess.

Requires GROUP privilege to stop other processes in the same group. Requires WORLD privilege to stop processes outside your group.

FORMAT

STOP *[process-name]*

PARAMETER

process-name

Requires that the process be in your group.

Specifies the name of the process to be deleted. The process name can have from 1 to 15 alphanumeric characters. The specified process must have the same group number in its user identification code (UIC) as the current process; you cannot use the process-name parameter to stop a process outside of your group. To stop a process outside of your group, you must use the qualifier /IDENTIFICATION=pid.

The process name cannot be used with the /IDENTIFICATION qualifier; if you use the /IDENTIFICATION qualifier, the process name is ignored. If you include neither the process-name parameter nor the /IDENTIFICATION qualifier with the STOP command, the image executing in the current process is terminated.

DESCRIPTION

The STOP command causes an abnormal termination of the image that is currently executing. If the image has declared any exit-handling routines, they are not given control. Use the EXIT command to terminate the image so that the exit-handling routines gain control.

Note that when an image has been interrupted by CTRL/Y and the RUN command is entered to execute another image, the interrupted image is terminated. However, in this case, exit-handling routines execute before the next image is run.

If the STOP command is executed from a noninteractive process (such as a batch job), the process terminates.

If you use CTRL/Y to interrupt a command procedure and then enter the STOP command, or if the STOP command is executed in a command procedure, all command levels are unstacked and control returns to command level 0 (DCL level with the $ prompt).

If you specify a process name or process identification code (PID), the STOP command terminates the image currently executing in the specified process and deletes the process. If the process is noninteractive, no notification of the deletion occurs and the log file for the job is not printed.

QUALIFIER

/IDENTIFICATION=pid

Specifies the system-assigned process identification code (PID). When you create a process with the RUN command, the RUN command displays the process identification code of the newly created process. /IDENTIFICATION can be used in place of the process name parameter.

You can omit any leading zeros in specifying the PID.

EXAMPLES

1

```
$ RUN MYPROG
    .
    .
    .
CTRL/Y
Interrupt
$ STOP
```

The RUN command in this example begins executing the image MYPROG. Subsequently, CTRL/Y interrupts the execution. The STOP command then terminates the image.

2

```
$ @TESTALL
    .
    .
    .
CTRL/Y
Interrupt
$ STOP
```

The @ (Execute Procedure) command in this example executes the procedure TESTALL.COM. CTRL/Y interrupts the procedure. The STOP command returns control to the DCL command interpreter.

3

```
$ RUN/PROCESS_NAME=LIBRA  LIBRA
%RUN-S-PROC_ID, identification of created process is 0013340D
    .
    .
    .
$ STOP LIBRA
```

The RUN command in this example creates a subprocess named LIBRA to execute the image LIBRA.EXE. Subsequently, the STOP command causes the image to exit and deletes the process.

4

```
$ ON ERROR THEN STOP
    .
    .
    .
```

In a command procedure, the ON command establishes a default action when any error occurs in the execution of a command or program. The STOP command stops all command levels. If this ON command is executed in a command procedure which in turn is executed from within another procedure, control does not return to the outer procedure, but to DCL command level 0.

STOP/CPU

Stops the specified secondary processor or processors in a VMS multiprocessing system. The /CPU qualifier is required.

Applies only to VMS multiprocessing systems. Requires change mode to kernel (CMKRNL) privilege.

FORMAT

STOP/CPU *[cpu-id,...]*

PARAMETER

cpu-id

Decimal value representing the identity of a processor in a VMS multiprocessing system. In a VAX 8300 system, for instance, the CPU ID is the VAXBI node number of the processor; in a VAX 8800, the CPU ID of the left processor is 1 and that of the right processor is 0. If you do not specify a CPU ID, the STOP/CPU command selects a processor in the current active set to stop.

DESCRIPTION

The STOP/CPU command removes a secondary processor from the active set in a VMS multiprocessing system. If the secondary processor is not executing a process when the STOP/CPU command is issued, it enters the STOPPED state. If the secondary is executing a process at the time, it continues to execute the current process until it becomes a candidate for rescheduling on another processor in the system. When this occurs, the secondary enters the STOPPED state.

The VMS operating system subjects a processor to a set of checks when it is the object of a STOP/CPU command. As a result, you may not be permitted to stop certain processors that are vital to the functioning of the system. In these cases, there is usually a process in the system that can execute only on the processor you intend to stop. You can determine this by issuing a SHOW CPU/FULL command. In unusual circumstances, you can bypass the checking mechanism by using the /OVERRIDE_CHECKS qualifier in the command.

The STOP/CPU command has no effect if its object processor is already in the STOPPED state when it is issued.

QUALIFIERS

/ALL

Stops all eligible secondary processors in the system's active set.

/OVERRIDE_CHECKS

Directs the STOP/CPU command to bypass a series of checks that determine whether the specified processor is eligible for removal from the active set.

EXAMPLES

1 `$ STOP/CPU`

The STOP/CPU command in this example selects a processor and removes it from the multiprocessing system's active set.

2 `$ STOP/CPU 4,7`

The STOP/CPU command in this example selects the processors with CPU IDs 04 and 07 and removes them from the multiprocessing system's active set.

3 `$ STOP/CPU/OVERRIDE_CHECKS 08`

The STOP/CPU/OVERRIDE_CHECKS command in this example unconditionally stops the processor with the CPU ID of 08 and removes it from active participation in the multiprocessing system.

4 `$ STOP/CPU/ALL`

The STOP/CPU/ALL command in this example stops all eligible secondary processors in the active set and removes them from the multiprocessing system.

STOP/QUEUE

The STOP/QUEUE command causes the specified execution queue to pause. All jobs currently executing in the queue are suspended (until the queue is restarted with the START/QUEUE command), and no new jobs can be initiated. The /QUEUE qualifier is required.

Requires OPER privilege or EXECUTE (E) access to the queue.

FORMAT

STOP/QUEUE *queue-name[:]*

PARAMETER

queue-name[:]
Specifies the name of the queue that you want to pause.

DESCRIPTION

The STOP/QUEUE command causes the specified queue to pause. All jobs currently executing in the queue are suspended. No new jobs can be initiated.

Use the START/QUEUE command to release the queue from the paused state. When you restart the queue, any jobs that were executing resume executing at the point where they left off, unless you use the /BACKWARD, /FORWARD, /SEARCH, or /TOP_OF_FILE qualifier to restart a print job at a different place.

For information about other STOP/QUEUE commands, see the following commands:

STOP/QUEUE/ABORT
STOP/QUEUE/ENTRY
STOP/QUEUE/MANAGER
STOP/QUEUE/NEXT
STOP/QUEUE/REQUEUE
STOP/QUEUE/RESET

EXAMPLES

1
```
$ STOP/QUEUE LPA0
```

The STOP/QUEUE command in this example halts the current print job in the queue LPA0 and places that queue in the paused state.

```
2  $ STOP/QUEUE TEXTBATCH
     .
     .
     .
   $ START/QUEUE/BLOCK_LIMIT=500 TEXTBATCH
```

The STOP/QUEUE command in this example halts all batch jobs that are currently executing on the queue TEXTBATCH and places that queue in the paused state. Later the START/QUEUE command releases the queue from the paused state. All the jobs that were halted resume processing, but the START/QUEUE command now limits any further jobs to 500 blocks or smaller.

STOP/QUEUE/ABORT

Aborts a job executing on a print or terminal queue, deletes it from the queue, and resumes execution of the other jobs in the queue. The /QUEUE qualifier is optional, but you must specify the /ABORT qualifier.

Requires OPER privilege, EXECUTE (E) access to the queue, or DELETE (D) access to the current job.

FORMAT

STOP/QUEUE/ABORT *queue-name[:]*

PARAMETER

queue-name
Specifies the name of the queue containing the job you want to stop.

DESCRIPTION

The STOP/QUEUE/ABORT command stops the job that is currently executing in a printer or terminal queue. The current job is deleted from the queue. (You do not specify a job entry number with /ABORT because printer and terminal queues can have only one current job at a time.) A user can issue this command to stop a job when that job is currently executing. Operators and system managers can use the command to abort the current job in a queue.

In general, the STOP/QUEUE/ABORT command is used to stop a print job that is no longer needed. Use the STOP/QUEUE/REQUEUE command to stop the current job and requeue it.

When you abort a print job with STOP/QUEUE/ABORT, the system attempts an orderly halt of the job. Assuming that the printing device is not malfunctioning, the print job completes the page that is currently printing. Then the job is removed from the queue. If the printer queue has been set up to put trailing pages at the end of jobs, a trailer page is printed after the current page is completed.

Note: If you accidentally enter the STOP/QUEUE/ABORT command for a malfunctioning queue, enter the STOP/QUEUE/RESET command to stop the queue in an orderly fashion.

EXAMPLE

```
$ STOP/QUEUE/ABORT LPA0
```

This example aborts the current print job on the queue LPA0. The next pending job in the queue begins to execute. Assuming there is no problem with the printer, the current page of the file completes printing. If the printer queue has been set up to output trailer pages, a trailer page is printed before the job is suspended.

STOP/QUEUE/ENTRY

Stops the currently executing job on the specified batch queue and resumes execution of the next pending job in the queue. The entry number is the number assigned to the job when it is submitted to the queue. (Use the DELETE/ENTRY command to stop an entry that is queued and awaiting execution.) The /QUEUE qualifier is optional, but you must specify the /ENTRY qualifier.

Requires OPER privilege, EXECUTE (E) access to the queue, or DELETE (D) access to the current job.

FORMAT

STOP/QUEUE/ENTRY=*entry-number queue-name[:]*

PARAMETER

queue-name
Specifies the name of the batch queue that contains the job you want to stop.

DESCRIPTION

The STOP/QUEUE/ENTRY command is used to stop a currently executing batch job. A user can enter the commands to stop a job when it is currently executing. Operators and system managers can use the commands to abort current batch jobs.

You can abort only a single current job with the STOP/QUEUE/ENTRY command, even if several jobs are currently executing in the batch queue. When you abort the batch job, the system attempts to stop the job in an orderly fashion, closing any open files and sending a message to the log file.

Use the STOP/QUEUE/REQUEUE command with the /ENTRY qualifier to stop the current batch job and requeue it.

Note: If you accidentally enter the STOP/QUEUE/ENTRY command for a malfunctioning queue, enter the STOP/QUEUE/RESET command to stop the queue in an orderly fashion.

EXAMPLE

```
$ STOP/QUEUE/ENTRY=365 SYS$BATCH
```

The STOP/QUEUE/ENTRY command in this example stops batch job number 365 currently executing on the SYS$BATCH queue and begins the next pending job in the queue.

STOP/QUEUE/MANAGER

Performs an orderly shutdown of the system job queue manager on the node from which the command is entered. The /QUEUE qualifier is optional, but you must specify the /MANAGER qualifier.

Requires both OPER and SYSNAM privileges.

FORMAT STOP/QUEUE/MANAGER

PARAMETERS

None.

DESCRIPTION

The STOP/QUEUE/MANAGER command carries out the following operations:

- Performs the STOP/QUEUE/NEXT operation for all output execution queues on the node from which the command is entered.
- Performs the STOP/QUEUE/ABORT operation for all current non-restartable jobs in all execution queues on the node from which the command is entered. Performs the STOP/QUEUE/REQUEUE operation for all current restartable jobs in all execution queues on that node.
- When all activity in all queues ceases, the STOP/QUEUE/MANAGER command closes the job queue manager file.

The STOP/QUEUE/MANAGER command is part of the SYS$SYSTEM:SHUTDOWN.COM procedure.

EXAMPLE

```
$ STOP/QUEUE/MANAGER
```

The STOP/QUEUE/MANAGER command in this example performs a shutdown of all queues on the node from which the command is entered.

STOP/QUEUE/NEXT

Stops the specified queue after all executing jobs have completed processing. No new jobs can be initiated; the START/QUEUE command restarts the queue. The /QUEUE qualifier is optional, but you must specify the /NEXT qualifier.

Requires OPER privilege or EXECUTE (E) access to the specified queue.

FORMAT

STOP/QUEUE/NEXT *queue-name[:]*

PARAMETER

queue-name[:]
Specifies the name of the queue that you want to stop.

DESCRIPTION

The STOP/QUEUE/NEXT command stops the queue completely after it allows any current jobs to complete execution. No new jobs can be initiated. Use the START/QUEUE command to restart the queue.

You should use the STOP/QUEUE/NEXT command before deassigning, deleting, merging, or requeuing a queue. Following this procedure allows all currently executing jobs to complete processing before changes are made to the queue.

If the printing device fails, enter the STOP/QUEUE/RESET command to stop the queue in an orderly fashion.

EXAMPLES

1
```
$ STOP/QUEUE/NEXT BATCH1
```

In this example, the STOP/QUEUE/NEXT command prepares to stop the queue BATCH1. All currently executing jobs are allowed to complete, but no new jobs are allowed to initiate. Once all current jobs have finished, the queue is stopped.

2
```
$ STOP/QUEUE/NEXT LPA0
$ SHOW QUEUE/ALL LPA0
Printer queue LPA0
$ DELETE/QUEUE LPA0
```

This example shows how to delete the printer queue LPA0. First, the STOP /QUEUE/NEXT command is entered, which stops the printer after the current job is printed. Then the SHOW QUEUE/ALL command is entered to ensure that no jobs are pending in the queue. The screen display shows that no jobs are pending. Finally, the DELETE/QUEUE command is entered to delete the printer queue LPA0.

STOP/QUEUE/REQUEUE

Stops the current job on the specified queue and requeues it for later processing. The queue does not stop; execution of the next pending job resumes. Print jobs that have been checkpointed resume printing at the checkpoint. Batch jobs containing SET RESTART_VALUE commands run those portions of the job that have not successfully completed. The /QUEUE qualifier is optional, but you must specify the /REQUEUE qualifier. If you are requeueing a job on a batch queue, you must specify the /ENTRY qualifier.

Requires OPER privilege, EXECUTE access to the queue or DELETE access to the current job.

FORMAT

STOP/QUEUE/REQUEUE*[=queue-name] queue-name[:]*
STOP/QUEUE/ENTRY*=entry-number/REQUEUE [=queue-name] queue-name[:]*

PARAMETER

queue-name
Specifies the name of the queue that contains the job you want to stop. When you also specify a queue name as a parameter for the /REQUEUE qualifier, the job is requeued to that queue.

DESCRIPTION

The STOP/QUEUE/REQUEUE command stops a currently executing job and requeues it for later processing. A user can enter the command to requeue a job when it is currently executing. Operators and system managers can use the command to abort or requeue current jobs.

If you include the queue name specifier with the /REQUEUE qualifier, STOP /QUEUE/REQUEUE transfers the current job to another queue. Otherwise, the job is requeued in the same queue.

You must use the /ENTRY qualifier with STOP/QUEUE/REQUEUE when entering the command for a batch queue.

The STOP/QUEUE/REQUEUE command causes the system to requeue the job for later execution in the queue. A print job that has been checkpointed resumes printing at the checkpoint where it left off, unless you enter the SET /QUEUE/ENTRY/NOCHECKPOINT command before the job is reinitiated. Batch jobs generally restart at the beginning. You can use SET RESTART_VALUE commands in a batch job to avoid rerunning portions of the job that have successfully completed.

You can use the STOP/QUEUE/ABORT command to stop a current print job without requeueing it. The STOP/QUEUE/ENTRY command stops a current batch job without requeueing it.

Note: If you accidentally issue the STOP/QUEUE/REQUEUE command for a malfunctioning queue, enter the STOP/QUEUE/RESET command to stop the queue in an orderly fashion.

QUALIFIERS

/ENTRY=entry-number

Used with batch queues to stop a currently executing batch job. The entry-number is the job entry number that was assigned to the job when it was submitted to the queue. The job entry number that you specify must match the job entry number of an executing job in order for the STOP/QUEUE /REQUEUE/ENTRY command to take effect.

You can only specify one entry number for each STOP/QUEUE/REQUEUE /ENTRY command.

/HOLD

Places the aborted job in a hold state for later release with the SET/QUEUE /ENTRY/RELEASE or SET QUEUE/ENTRY/NOHOLD command. (Use DELETE/ENTRY to delete a job in the hold state.)

/PRIORITY=n

Requires OPER or ALTPRI privileges to raise the priority value above the value of the SYSGEN parameter MAXQUEPRI.

Changes the priority of the requeued job. The n parameter can be from 0 to 255; the default value of the n parameter is the same as the priority value that the job had when it was stopped.

Generally, the /PRIORITY qualifier is used to lower the priority of a job, which ensures the job will run when the queue contains no other jobs. No privilege is needed to set the priority lower than the MAXQUEPRI value.

EXAMPLES

1
```
$ STOP/QUEUE/REQUEUE=LPB0 LPA0
```

In this example, the current print job on queue LPA0 is stopped and requeued to queue LPB0. If the print job has been checkpointed, printing resumes on LPB0 where the job stopped on LPA0.

2
```
$ STOP/QUEUE/REQUEUE/HOLD LPA0
.
.
.
$ SET QUEUE/ENTRY=254/RELEASE
```

In this example, the job currently printing on LPA0 is suspended and placed in the hold state. Later, when the SET QUEUE/ENTRY command is entered with the /RELEASE qualifier, the job state changes from holding to pending and remains there until the job can begin printing. If the print job has been checkpointed, printing resumes where the job stopped.

3
```
$ STOP/QUEUE/REQUEUE/ENTRY=758 SYS$BATCH
```

In this example, batch job number 758 is stopped and requeued for later processing on SYS$BATCH. If the batch job has been programmed with appropriate SET RESTART_VALUE commands, those portions of the job that have successfully completed are not rerun.

STOP/QUEUE/RESET

Abruptly stops the queue and returns control to the system. Any jobs currently executing are stopped immediately. The START/QUEUE command restarts the queue. Current jobs that can be restarted (all print jobs and any batch jobs submitted with the /RESTART qualifier) are requeued for processing. Current jobs that cannot be restarted are aborted and must be resubmitted for processing. The /QUEUE qualifier is optional, but you must specify the /RESET qualifier.

Requires OPER privilege or EXECUTE (E) access to the specified queue.

FORMAT

STOP/QUEUE/RESET *queue-name[:]*

PARAMETER

queue-name[:]
Specifies the name of the queue you want to reset.

DESCRIPTION

The STOP/QUEUE/RESET command stops the queue as soon as the system receives the command. The queue manager requests termination for all executing jobs, but aborts or requeues executing jobs without waiting for termination status to be received.

Use the START/QUEUE command to restart the queue. Current jobs that can be restarted are requeued for processing. Current jobs that cannot be restarted are aborted and must be resubmitted for processing. (Print jobs are restartable by default. Use the SUBMIT/RESTART command to make a batch job restartable.)

EXAMPLES

1
```
$ STOP/QUEUE/RESET LPA0
```

The STOP/QUEUE/RESET command in this example abruptly stops the printer queue LPA0. The current print job stops immediately.

2
```
$ STOP/QUEUE/RESET TEXBATCH
```

The STOP/QUEUE/RESET command in this example stops the TEXBATCH queue. Any current job that was submitted with the /RESTART qualifier is requeued for processing when the queue is restarted. Current jobs that did not specify /RESTART must be resubmitted to the queue.

SUBMIT

Enters one or more command procedures in a batch job queue.

Requires EXECUTE (E) access to the queue, WRITE (W) access to the queue, or OPER privilege. If you include a node name with the file specification parameter, you must use the /REMOTE qualifier.

FORMAT

SUBMIT *file-spec[,...]*

PARAMETER

file-spec[,...]

Specifies the name of a file containing a command procedure. Wildcard characters are allowed. The default file type is COM. If you specify two or more files, separate the file specifications either with commas or plus signs. In either case, the files are processed serially as a single batch job.

If a node name is specified, the /REMOTE qualifier must also be specified.

DESCRIPTION

The SUBMIT command enters command procedures in the batch job queue. Each command procedure is contained in a file specified with the SUBMIT command. If you specify two or more files in the command line, the system considers the entries as a single job.

The system assigns a unique job entry number to each batch job in the queue. When you submit a batch job, by default the system displays both the job entry number it has assigned to the job, the name of the batch job queue in which your job has been entered, and the current job status, for example, executing, pending, or holding.

Once a batch job has been queued, the version of the file submitted is processed, even if a newer version of the file is created before the batch job runs.

When the system executes a command procedure submitted to a batch job queue, it creates a detached process to execute the commands. This process receives your disk and directory defaults and the same resource quotas and privileges that were given to your interactive process when you logged in.

Batch Job Output

When you submit a command procedure for processing by the SUBMIT command, all output from the command procedure is written to a log file. By default, the log file is named job_name.log with "job_name" being the name of the first command procedure file in the job. The log file also can be named using the /NAME qualifier or the /LOG_FILE qualifier. The log file is written to the directory defined by the logical name SYS$LOGIN in the UAF, unless you specify otherwise with the /LOG_FILE qualifier.

After the batch job finishes, the system queues the log file to the SYS$PRINT queue and deletes the file after it has printed (unless you have specified /KEEP). If you do not want the log file printed, use the /NOPRINT qualifier. When you stop a batch job using the DELETE/ENTRY, STOP

/IDENTIFICATION, or STOP/QUEUE/ENTRY command, the log file is not queued for printing.

If multiple procedures are submitted, the job terminates as soon as any procedure exits with an error or severe (fatal) error status.

QUALIFIERS

/AFTER=time
/NOAFTER

Requests that the job be held until after a specific time. If the specified time has already passed, the job is processed immediately.

Time can be specified as either an absolute time or as a combination of absolute and delta times. See the *VMS DCL Concepts Manual* for complete information on specifying time values.

In a VAXcluster, a batch job submitted to execute at a specific time may begin execution a little before or after the requested time. This occurs when the clocks of the member systems in the VAXcluster are not synchronized. For example, a job submitted using the DCL command SUBMIT/AFTER=TOMORROW may execute at 23:58 relative to the host system's clock.

This problem can occur in a cluster even if a job is run on the same machine from which it was submitted, because the redundancy built into the batch/print system allows more than one job controller in the cluster to receive a timer AST for the job and, thus, to schedule it for execution. Moreover, this behavior is exacerbated if the batch job immediately resubmits itself to run the next day using the same SUBMIT command. This can result in having multiple instances of the job executing simultaneously because TOMORROW (after midnight) may be only a minute or two in the future.

A solution to this problem is to place the SUBMIT command in a command procedure that begins with a WAIT command, where the *delta-time* specified in the WAIT command is greater than the maximum difference in time between any two systems in the cluster. Use the SHOW TIME command on each system to determine this difference in time. Use the SET TIME command with the /CLUSTER qualifier to synchronize clocks on the cluster.

/BACKUP
/NOBACKUP

Modifies the time value specified with the /BEFORE or /SINCE qualifier. /BACKUP selects files according to the dates of their most recent backups. This qualifier is incompatible with the other qualifiers that also allow you to select files according to time attributes: /CREATED, /EXPIRED, and /MODIFIED. If you specify none of these four time qualifiers, the default is /CREATED.

/BEFORE[=time]
/NOBEFORE

Selects only those files dated prior to the specified time. You can specify time as an absolute time, as a combination of absolute and delta times, or as one of the following keywords: TODAY (default), TOMORROW, or YESTERDAY. Specify one of the following qualifiers with /BEFORE to indicate the time attribute to be used as the basis for selection: /BACKUP, /CREATED (default), /EXPIRED, or /MODIFIED.

See the *VMS DCL Concepts Manual* for complete information on specifying time values.

/BY_OWNER[=uic]
/NOBY_OWNER

Selects only those files whose owner user identification code (UIC) matches the specified owner UIC. The default UIC is that of the current process.

Specify the UIC using standard UIC format as described in Section 8.1 of the *VMS DCL Concepts Manual*.

/CHARACTERISTICS=(characteristic[,...])

Specifies one or more characteristics that the execution queue must possess for the job to run. The characteristics you specify must be a subset of the characteristics associated with the queue that executes the job. Otherwise, the job remains pending until the queue characteristics are changed, or until you delete the entry with the DELETE/ENTRY command. By default, the job runs if no characteristics are specified.

If you specify only one characteristic, you can omit the parentheses. Codes for characteristics can be either names or values from 0 to 127 and are installation-defined. Use the SHOW QUEUE/CHARACTERISTICS command to see which characteristics have been defined for your system. Use the SHOW QUEUE command with the /FULL qualifier to see which characteristics are available on a particular queue.

/CLI=filename

Specifies the command language interpreter (CLI) to be used to process the job. The file specification assumes the device name SYS$SYSTEM: and the file type EXE (SYS$SYSTEM:filename.EXE). The default CLI is that defined in the user authorization file.

/CONFIRM
/NOCONFIRM (default)

Controls whether a request is issued before each SUBMIT operation to confirm that the operation should be performed on that file. The following responses are valid:

YES	NO	QUIT
TRUE	FALSE	CTRL/Z
1	0	ALL
	RET	

You can use any combination of uppercase and lowercase letters for word responses. Word responses can be abbreviated to one or more letters (for example, T, TR, or TRU for TRUE), but these abbreviations must be unique. Affirmative answers are YES, TRUE, and 1. Negative answers are NO, FALSE, 0, and <RET>. QUIT or CTRL/Z indicates that you want to stop processing the command at that point. When you respond with ALL, the command continues to process, but no further prompts are given. If you type a response other than one of those in the list, DCL issues an error message and redisplays the prompt.

/CPUTIME=keyword

Defines a CPU time limit for the batch job. Time can be specified as delta time, 0, NONE, or INFINITE (see the *VMS DCL Concepts Manual*). When you need less CPU time than authorized, use the /CPUTIME qualifier to override the base queue value established by the system manager or the value authorized for you in the user authorization file (UAF). Both the value 0 and the keyword INFINITE allow unlimited CPU time; the keyword NONE defaults to your user authorization file (UAF) value or to the limit specified on the queue. Note that you cannot request more time than permitted by the base queue limits or by your own UAF.

/CREATED (defau)
/NOCREATED

Modifies the time value specified with the /BEFORE or /SINCE qualifier. /CREATED selects files based on their dates of creation. This qualifier is incompatible with the other qualifiers that also allow you to select files according to time attributes: /BACKUP, /EXPIRED, and /MODIFIED. If you specify none of these four time qualifiers, the default is /CREATED.

/DELETE
/NODELETE (default)

Positional qualifier.

Controls whether files are deleted after processing. If you specify the /DELETE qualifier after the SUBMIT command name, all files in the job are deleted after processing. If you specify the /DELETE qualifier after a file specification, only that file is deleted after it is processed.

For /DELETE to work, the protection code on the input files must allow D (delete) access to the user identification code (UIC) of the user submitting the job.

/EXCLUDE=(file-spec[,...])
/NOEXCLUDE

Excludes the specified files from the SUBMIT operation. You can include a directory but not a device in the file specification. Wildcard characters are allowed in the file specification. However, you cannot use relative version numbers to exclude a specific version. If you provide only one file specification, you can omit the parentheses.

/EXPIRED
/NOEXPIRED

Modifies the time value specified with the /BEFORE or /SINCE qualifier. /EXPIRED selects files according to their expiration dates. (The expiration date is set with the SET FILE/EXPIRATION_DATE command.) The /EXPIRED qualifier is incompatible with the other qualifiers that also allow you to select files according to time attributes: /BACKUP, /CREATED, and /MODIFIED. If you specify none of these four time qualifiers, the default is /CREATED.

/HOLD
/NOHOLD (default)

Controls whether or not the job is made available for immediate processing. The /HOLD qualifier holds the job until it is released by the /NOHOLD qualifier or by the /RELEASE qualifier of the SET QUEUE/ENTRY command.

/IDENTIFY (default)
/NOIDENTIFY
Displays the queue name and job-entry number of the job when it is queued.

/KEEP
/NOKEEP
Controls whether the log file is deleted after it is printed; /NOKEEP is the default unless /NOPRINTER is specified.

/LOG_FILE[=file-spec]
/NOLOG_FILE
Names the log file. The default log file name is *job-name.log*. No wildcards are allowed in the file specification.

When you use the /LOG_FILE qualifier, the system writes the log file to the file you specify. If you use /NOLOG_FILE, no log file is created. By default, a log file is kept and written to *SYS$LOGIN:job_name.log*.

You can use the /LOG_FILE qualifier to write the log file to a different device. Logical names in the file specification are translated in the context of the process that submits the job. The process executing the batch job must have access to the device on which the log file will reside.

If you omit the /LOG_FILE qualifier and specify the /NAME qualifier, the log file is written to a file having the same file name as that specified by the /NAME qualifier with the file type LOG. When you omit the /LOG_FILE qualifier, the job-name value used with /NAME must be a valid file name.

/MODIFIED
/NOMODIFIED
Modifies the time value specified with the /BEFORE or /SINCE qualifier. /MODIFIED selects files according to the dates on which they were last modified. This qualifier is incompatible with the other qualifiers that also allow you to select files according to time attributes: /BACKUP, /CREATED, and /EXPIRED. If you specify none of these four time modifiers, the default is /CREATED.

/NAME=job-name
Names the job (and possibly the batch job log file). The job name must be 1 to 39 alphanumeric characters. If characters other than alphanumerics, underscores, or dollar signs are used in the name, enclose the name in quotation marks. The default job name is the name of the first (or only) file in the job.

If the /LOG_FILE qualifier is omitted, the job-name value must be a valid file name. The job name is displayed by the SHOW QUEUE command.

/NOTIFY
/NONOTIFY (default)
Controls whether a message is broadcast to your terminal when the job is completed or aborted.

/PARAMETERS=(parameter[,...])
Provides the values of up to 8 optional parameters (equated to the symbols P1 through P8, respectively, for each command procedure in the job). The symbols are local to the specified command procedure.

Commas delimit individual parameters. If you specify only one parameter, you can omit the parentheses.

If the parameter contains spaces, special characters, or lowercase characters, enclose it in quotation marks. The size of the parameter can be from 1 to 255 characters.

/PRINTER[=queue-name](default)
/NOPRINTER

Queues the job log file for printing when your job is completed. /PRINTER allows you to specify a particular print queue; the default print queue is SYS$PRINT. If you specify /NOPRINTER, /KEEP is assumed.

/PRIORITY=n

Requires OPER or ALTPRI to specify a priority greater than the value of the SYSGEN parameter MAXQUEPRI.

Specifies the job-scheduling priority for the batch job with respect to other jobs in the same queue. The value of *n* is an integer in the range of 0 through 255, where 0 is the lowest priority and 255 is the highest.

The default value is the value of the SYSGEN parameter DEFQUEPRI. No privilege is needed to set the priority lower than the MAXQUEPRI value.

The /PRIORITY qualifier has no effect on the job's process execution priority. The job's process execution priority is determined by the base priority attribute of the INITIALIZE/QUEUE/BASE_PRIORITY command.

/QUEUE=queue-name[:]

Identifies the batch queue on which the job is entered. The default queue is SYS$BATCH.

/REMOTE

Indicates that the file resides on the remote node indicated and executes on the remote node. When using /REMOTE, the node name *must* be included in the file specification.

Note that, unlike the local case, multiple command procedures queued by a single SUBMIT/REMOTE command are considered separate jobs.

Only the following qualifiers (which affect file selection) may be specified with /REMOTE: /BACKUP, /BEFORE, /BY_OWNER, /CONFIRM, /CREATED, /EXCLUDE, /EXPIRED, /MODIFIED, and /SINCE.

/RESTART
/NORESTART (default)

Indicates whether or not the job restarts after a system failure or after a STOP /QUEUE/REQUEUE command.

/SINCE[=time]
/NOSINCE

Selects only those files dated after the specified time. You can specify time as an absolute time, a combination of absolute and delta times, or as one of the following keywords: TODAY (default), TOMORROW, or YESTERDAY. Specify one of the following qualifiers with /BEFORE to indicate the time attribute to be used as the basis for selection: /BACKUP, /CREATED (default), /EXPIRED, or /MODIFIED.

See the *VMS DCL Concepts Manual* for complete information on specifying time values.

/USER=username

Requires CMKRNL privilege and R (read) access to the user authorization file (UAF).

Allows you to submit a job on behalf of another user. The job runs exactly as if that user had submitted it. The job runs under that user's user name and UIC and accounting information is logged to that user's account. By default, the user identification comes from the requesting process. The username parameter can be any user name that is validated on your system.

/WSDEFAULT=n

Defines a working set default for a batch job; the /WSDEFAULT qualifier overrides the working set size authorized by the system manager or specified in the user authorization file. Possible values of *n* are a positive integer in the range 1 through 65,535, 0, or the keyword NONE. Specify 0 or NONE if you want the working set value to default to either your UAF value or the working set default specified on the queue. You cannot request a higher value than your default.

/WSEXTENT=n

Defines a working set extent for the batch job; the /WSEXTENT qualifier overrides the working set extent authorized by the system manager or specified in the user authorization file. Possible values of *n* are a positive integer in the range 1 through 65,535, 0, or the keyword NONE. Specify 0 or NONE if you want the working set extent to default to either your UAF value or the working set extent specified on the queue. You cannot request a higher value than your default.

/WSQUOTA=n

Defines a working set page size (working set quota) for the batch job; the /WSQUOTA qualifier overrides the value established by the system manager or the value authorized in the user authorization file. Possible values of *n* are a positive integer in the range 1 through 65,535, 0, or the keyword NONE. Specify 0 or NONE if you want the working set quota to default to either your UAF value or the working set quota specified on the queue. You cannot request a higher value than your default.

EXAMPLES

1
```
$ SUBMIT/AFTER=16:30 TRANSLATE
  Job TRANSLATE (queue SYS$BATCH, entry 1401) holding until 31-DEC-1988 16:30
```

In this example, the command procedure TRANSLATE.COM is submitted to SYS$BATCH. The procedure is executed at 16:30 or later. When the batch job completes, the log file TRANSLATE.LOG is queued for printing and deleted.

2
```
$ SUBMIT /PARAMETERS=(TXT,DOC,MEM) BACKUP, -
$_AVERAGE, RUNMASTER
  Job AVERAGE (queue SYS$BATCH, entry 416) pending
```

In this example, the SUBMIT command enters three command procedures in a single job. The job is given three parameters: P1 is equated to the string TXT, P2 to the string DOC, and P3 to the string MEM. After the procedure BACKUP.COM is executed, the procedures AVERAGE.COM and RUNMASTER.COM are executed.

3
```
$ SUBMIT/NAME=BATCH24/HOLD TESTALL
  Job BATCH24 (queue SYS$BATCH, entry 467) holding
```

In this example, the SUBMIT command enters the procedure TESTALL.COM as a batch job and specifies that the job be held for later processing. The job is not released until the SET QUEUE/ENTRY/RELEASE command is entered. The /NAME qualifier requests that the batch job be identified as BATCH24.

4
```
$ DEFINE JUNE WORKZ:[JONES]ANNUAL_REPORT.COM
$ SUBMIT JUNE
Job JUNE (queue SYS$BATCH, entry 229) started on ZOO_BATCH
```

In this example, the logical name JUNE is created and equated to ANNUAL_REPORT.COM with the DEFINE command. Using the logical name JUNE, the user submits ANNUAL_REPORT.COM to the batch queue. Note that the system translates the logical name JUNE to ANNUAL_REPORT.COM before ANNUAL_REPORT.COM is submitted to the batch queue. Also, the log file produced is named ANNUAL_REPORT.COM rather than JUNE.COM.

Note also that the job is submitted to the generic queue SYS$BATCH, but runs on the execution queue ZOO_BATCH.

SUBROUTINE

Defines the beginning of a subroutine in a command procedure. The SUBROUTINE command must be the first executable statement in a subroutine. For more information about the SUBROUTINE command, refer to the description of the CALL command.

FORMAT SUBROUTINE

SYNCHRONIZE

Holds the process issuing the command until the specified batch job completes execution.

FORMAT

SYNCHRONIZE *[job-name]*

PARAMETER

job-name

Specifies the name of the job as specified in the SUBMIT command. You can specify only job names that are associated with your user name. (A job is associated with the user name of the process that submits it.)

If you have more than one job with the same name, the process is synchronized with the last job submitted. To specify a job that does not have a unique name, use the /ENTRY qualifier to specify the job entry number. If you use the /ENTRY qualifier and if you also specify a job name, the job name is ignored.

DESCRIPTION

The SYNCHRONIZE command provides job synchronization by placing a process in a wait state until the specified job completes. If the specified job is not currently in the system, the SYNCHRONIZE command issues an error message.

When a job specified in a SYNCHRONIZE command completes, the process is released from the wait state. The completion status for the SYNCHRONIZE command is the same as the completion status of the last command executed in the job.

QUALIFIERS

/ENTRY=entry-number

Identifies the job by the system-assigned entry number. By default, the system displays the entry number when it successfully queues a job for execution; the entry number of a job is also displayed when you enter the SHOW QUEUE command. You must specify either the job name or the /ENTRY qualifier. If you specify both the job-name parameter and the /ENTRY qualifier, the job name is ignored.

/QUEUE=queue-name[:]

Names the queue containing the job. The default is SYS$BATCH.

EXAMPLES

1

```
$ SUBMIT/NAME=PREP FORMAT/PARAMETERS=(SORT,PURGE)
  Job PREP (queue SYS$BATCH, entry 219) started on queue SYS$BATCH
$ SUBMIT PHASER
```

In this example, the first SUBMIT command submits the command procedure FORMAT.COM for execution and names the job PREP. The second SUBMIT command queues the procedure PHASER.COM. The procedure PHASER.COM contains the following line:

```
$ SYNCHRONIZE PREP
```

When this line is processed, the system verifies whether the job named PREP is currently executing in SYS$BATCH. (SYS$BATCH is the default queue for the SYNCHRONIZE command.) The procedure PHASER is forced to wait until the job PREP completes execution.

2

```
$ SUBMIT/NAME=TIMER       COMP.COM
  Job TIMER (queue SYS$BATCH, entry 214) started on queue SYS$BATCH
$ SYNCHRONIZE /ENTRY=214
```

In this example, a batch job named TIMER is submitted. Then the SYNCHRONIZE command is entered interactively. This command places the interactive process in a wait state until entry number 214 (TIMER) completes. You cannot enter subsequent commands from your terminal until the SYNCHRONIZE command completes and your process is released from the wait state.

TYPE

Displays the contents of a file or group of files on the current output device.

FORMAT

TYPE *file-spec[,...]*

PARAMETER

file-spec[,...]

Specifies one or more files to be displayed. If you specify a file name and not a file type, the file type defaults to LIS. The TYPE command displays all files that satisfy the file description.

Wildcard characters are allowed in place of the directory name, file name, file type, or file version number field. Either commas or plus signs can be used to separate two or more files. The files are displayed in the order listed.

DESCRIPTION

When the TYPE command displays output, you can control the display in the following ways:

- Use CTRL/C to stop the TYPE command for the current file that is being displayed. If you specified only one file with your TYPE command, or if TYPE is displaying the last file in the list, pressing CTRL/C cancels the TYPE command. If you have specified more than one file with the TYPE command, pressing CTRL/C causes TYPE to display the next file in the list.
- Use CTLR/S to temporarily suspend the output. Use CTRL/Q to resume the output display at the point of interruption.
- Use CTRL/O to suppress the display but not suspend the command processing. If you press CTRL/O again before the TYPE command terminates, output resumes at the current point in command processing. However, if you press CTRL/O when the TYPE command is displaying files in a list, the TYPE command suppresses typing the current file and begins typing the next file in the list. This behavior is an exception to normal CTRL/O processing.
- Use CTRL/Y to interrupt the command execution. You can enter the CONTINUE command after pressing CTRL/Y to resume displaying the files where the interruption took place, provided you have not entered an intervening command that calls up a new image. If you press CTRL/Y to stop command execution entirely, you can enter the EXIT command (or any other DCL command that activates an image) to run down the image.

In addition, the /PAGE qualifier may be used to display text one screen at a time.

TYPE opens the specified file with shared read and write access. Therefore, any file that has its attributes set to shared write is displayed, even if it is currently opened by another user.

You also can use the TYPE command to execute a command procedure on a remote node. This is useful on clusters, for example, when you want to display the status of cluster-wide services, such as queues, or when you want to display the users logged on to other nodes on the cluster. A sample command procedure follows:

```
$ ! SHOWUSERS.COM
$ if f$mode() .eqs. "NETWORK" then define/user sys$output sys$net
$ show users
```

This command procedure can be used with the TYPE command to display at the user's local node the users logged on to the remote node where the command procedure resides.

Specify the command procedure as a parameter to the TYPE command as follows:

```
$ TYPE node_name::"TASK=command_procedure"
```

where:

node_name	is the name of the remote node on which the command procedure resides.
command_procedure	is the filename of the command procedure to be run.

This form of the command finds the command procedure in the default DECnet account of the remote node.

To execute a command procedure in the SYS$LOGIN directory of a particular account use an Access Control String in the command, as follows:

```
$ TYPE node_name"user_name password"::"TASK=command_procedure"
```

where:

user_name	is the user name of the account on the remote node.
password	is the password of the account on the remote node.

QUALIFIERS

/BACKUP

Modifies the time value specified with the /BEFORE or /SINCE qualifier. /BACKUP selects files according to the dates of their most recent backups. This qualifier is incompatible with the other qualifiers that also allow you to select files according to time attributes: /CREATED, /EXPIRED, and /MODIFIED. If you specify none of these four time qualifiers, the default is /CREATED.

/BEFORE[=time]

Selects only those files dated prior to the specified time. You can specify time as an absolute time, as a combination of absolute and delta times, or as one of the following keywords: TODAY (default), TOMORROW, or YESTERDAY. Specify one of the following qualifiers with /BEFORE to indicate the time attribute to be used as the basis for selection: /BACKUP, /CREATED (default), /EXPIRED, or /MODIFIED.

See Section 1.4 of the *VMS DCL Concepts Manual* for complete information on specifying time values.

/BY_OWNER[=uic]

Selects only those files whose owner user identification code (UIC) matches the specified owner UIC. The default UIC is that of the current process.

Specify the UIC using standard UIC format as described in Section 8.1 of the *VMS DCL Concepts Manual*.

/CONFIRM
/NOCONFIRM (default)

Controls whether a request is issued before each TYPE operation to confirm that the operation should be performed on that file. The following responses are valid:

YES	NO	QUIT
TRUE	FALSE	CTRL/Z
1	0	ALL
	RET	

You can use any combination of uppercase and lowercase letters for word responses. Word responses can be abbreviated to one or more letters (for example, T, TR, or TRU for TRUE), but these abbreviations must be unique. Affirmative answers are YES, TRUE, and 1. Negative answers are NO, FALSE, 0, and <RET>. QUIT or CTRL/Z indicates that you want to stop processing the command at that point. When you respond with ALL, the command continues to process, but no further prompts are given. If you type a response other than one of those in the list, DCL issues an error message and redisplays the prompt.

/CREATED (default)

Modifies the time value specified with the /BEFORE or /SINCE qualifier. /CREATED selects files based on their dates of creation. This qualifier is incompatible with the other qualifiers that also allow you to select files according to time attributes: /BACKUP, /EXPIRED, and /MODIFIED. If you specify none of these four time qualifiers, the default is /CREATED.

/EXCLUDE=(file-spec[,...])

Excludes the specified files from the TYPE operation. You can include a directory but not a device in the file specification. Wildcard characters are allowed in the file specification. However, you cannot use relative version numbers to exclude a specific version. If you provide only one file specification, you can omit the parentheses.

/EXPIRED

Modifies the time value specified with the /BEFORE or /SINCE qualifier. /EXPIRED selects files according to their expiration dates. (The expiration date is set with the SET FILE/EXPIRATION_DATE command.) The /EXPIRED qualifier is incompatible with the other qualifiers that also allow you to select files according to time attributes: /BACKUP, /CREATED, and /MODIFIED. If you specify none of these four time qualifiers, the default is /CREATED.

/MODIFIED

Modifies the time value specified with the /BEFORE or /SINCE qualifier. /MODIFIED selects files according to the dates on which they were last modified. This qualifier is incompatible with the other qualifiers that also allow you to select files according to time attributes: /BACKUP, /CREATED,

and /EXPIRED. If you specify none of these four time modifiers, the default is /CREATED.

/OUTPUT[=file-spec]
/NOOUTPUT

Controls where the output of the command is sent. If you specify /OUTPUT=file-spec, the output is sent to the specified file, rather than to the current output device, SYS$OUTPUT. If you do not enter the qualifier, or if you enter /OUTPUT without a file specification, the output is sent to SYS$OUTPUT.

If you enter /OUTPUT with a partial file specification (for example, /OUTPUT=[JONES]), TYPE is the default file name and LIS the default file type. If you enter a file specification, it may not include any wildcard characters.

If you enter /NOOUTPUT, output is suppressed.

The /OUTPUT qualifier is incompatible with the /PAGE qualifier.

/PAGE
/NOPAGE (default)

Controls whether output from the TYPE command is displayed one screen at a time. If more than one file has been requested, CTRL/Z may be used to cancel the display of the current file and to continue with the next file.

The /PAGE qualifier is incompatible with the /OUTPUT qualifier.

/SINCE[=time]

Selects only those files dated after the specified time. You can specify time as an absolute time, a combination of absolute and delta times, or as one of the following keywords: TODAY (default), TOMORROW, or YESTERDAY. Specify one of the following qualifiers with /BEFORE to indicate the time attribute to be used as the basis for selection: /BACKUP, /CREATED (default), /EXPIRED, or /MODIFIED.

See Section 1.4 of the *VMS DCL Concepts Manual* for complete information on specifying time values.

EXAMPLES

1
```
$ TYPE  COMMON.DAT
```

In this example, the TYPE command requests that the file COMMON.DAT be displayed at the terminal.

2
```
$ TYPE *.DAT
This is the first line in the file AA.DAT.
   .
   .
   .
^O
This is the first line in the file BB.DAT.
   .
   .
   .
CTRL/Y
Interrupt

$ STOP
```

In this example, the TYPE command contains a wildcard character in place of the file name. All files with file types of DAT are scheduled for display. When CTRL/O is pressed, output of the current file stops and the TYPE command begins displaying the next file. CTRL/Y interrupts the command; the STOP command terminates the TYPE command.

3
```
$ TYPE LETTER*.MEM
December 31, 1988
   .
   .
   .
CTRL/Y
Interrupt

$ SHOW TIME
 31-DEC-1988 15:48:07
$ CONTINUE
Sincerely yours,
   .
   .
   .
```

In this example, the TYPE command displays all files whose names begin with the word LETTER and have the file type MEM. While the files are being displayed, the user presses CTRL/Y to interrupt the TYPE operation and display the time. After entering the SHOW TIME command, the user enters the CONTINUE command to resume the TYPE operation.

4
```
$ TYPE/OUTPUT=SAVETEXT.TXT *.TXT
```

In this example, the TYPE command writes all TXT files in your default directory to a file called SAVETEXT.TXT (also in your default directory).

5
```
$ TYPE MEXICO::NOTICE.TEXT/OUTPUT=TEMP.TEXT
```

In this example, the TYPE command requests that the file NOTICE.TEXT at remote node MEXICO be written to the output file TEMP.TEXT on the local node, rather than to SYS$OUTPUT.

6

```
$ TYPE SECSSYS"FILES OFFICEFIL"::"TASK=SHOWUSERS"

          VAX/VMS Interactive Users
           09-DEC-1988 17:20:13.30
    Total number of interactive users = 5
 Username     Process Name      PID     Terminal
 KAITLIN      Sec1           00536278   TXA1:
 LINDSEY      Sec2           00892674   VTA2:
 ALYSON       Sec3           00847326   TXA3:
 KIM          Sec4           02643859   RTA1:
 GERARD       System Mangr   00007362   VTA1:
```

In this example, the TYPE command executes the command procedure SHOWUSERS.COM found in the SYS$LOGIN directory of user FILES on remote node SECSSYS. The output of the TYPE command then is displayed at the local node.

UNLOCK

Makes an improperly closed file accessible.

FORMAT

UNLOCK *file-spec[,...]*

PARAMETER

file-spec[,...]

Specifies the name of the file to be unlocked. Wildcard characters are allowed. If you include two or more file specifications, separate them with either commas or plus signs.

QUALIFIERS

/CONFIRM
/NOCONFIRM (default)

For each file being unlocked, displays a query to which you must respond Y (YES) or T (TRUE) to unlock the file. Any other response aborts the unlock operation.

/LOG
/NOLOG (default)

Controls whether the UNLOCK command displays the file specification of each file being unlocked.

EXAMPLES

1

```
$ TYPE TST.OUT
%TYPE-E-OPENIN, error opening  DISK1:[STEVE]TST.OUT;3 as input
-SYSTEM-W-FILELOCKED, file is deaccess locked
$ UNLOCK TST.OUT
$ TYPE TST.OUT
```

In this example, the request to type the output file TST.OUT returns an error message indicating that the file is locked. The UNLOCK command unlocks it. Then the TYPE command is reentered to display the contents of the file.

2

```
$ UNLOCK NODE3::DISKO:[LISTS]MAILLIST3.LIS
$ COPY NODE3::DISKO:[LISTS]MAILLIST3.LIS *.*
```

In this example, the user needs a copy of the file MAILLIST3.list, which is locked on remote NODE3. Enter the UNLOCK command first and then copy the file to your current node, disk, and directory.

WAIT

Puts your process into a wait state for the specified amount of time. The WAIT command is used in a command procedure to delay processing of either the procedure itself or a set of commands in the procedure.

FORMAT

WAIT *delta-time*

PARAMETER

delta-time

Specifies a delta time interval in the following format. (A delta time is an offset from the current time to a time in the future.)

```
hour:minute:second.hundredth
```

The fields on the format line indicate the following:

hour	An integer in the range 0 through 59
minute	An integer in the range 0 through 59
second	An integer in the range 0 through 59
hundredth	An integer in the range 0 through 99.

The colons and period are required delimiters; also, the delta time must begin with the number of hours and not a colon. Note that the days field, usually included in the delta time format, must be omitted here.

See Section 1.4 of the *VMS DCL Concepts Manual* for more information on specifying delta time values.

Note that if you enter the WAIT command interactively, you are not prompted for a time value. However, in order for the command to have any effect, you must supply a time value.

DESCRIPTION

If you enter the WAIT command interactively, your current process is placed in a wait state and you cannot enter any more commands until the waiting period is over. (You can, however, receive unsolicited messages from other processes.) Press CTRL/C or CTRL/Y to restore normal terminal interaction.

EXAMPLE

```
$ LOOP:
$ RUN ALPHA
$ WAIT 00:10
$ GOTO LOOP
```

In this example, the command procedure executes the program image ALPHA. After the RUN command executes the program, the WAIT command delays execution of the GOTO command for 10 minutes. Note that 00 is specified for the number of hours, because the time specification cannot begin with a colon. After 10 minutes, the GOTO command executes, and

the procedure transfers control to the label LOOP and executes the program ALPHA again. The procedure loops until it is interrupted or terminated.

If the procedure is executed interactively, terminate it by pressing CTRL/C or CTRL/Y and entering the STOP command or another DCL command that runs a new image in the process. If the procedure is executed in a batch job, enter the DELETE/ENTRY command to terminate it.

WRITE

Writes the specified data as one record to an open file specified by a logical name.

All qualifiers must precede all *data-item* expressions.

FORMAT

WRITE *logical-name expression[,...]*

PARAMETERS

logical-name

Specifies the logical name assigned to the output file. Use the logical name assigned by the OPEN command. In interactive mode, specify the process-permanent files identified by the logical names SYS$INPUT, SYS$OUTPUT, SYS$ERROR, and SYS$COMMAND. (The OPEN command assigns a logical name to a file and places the name in the process logical name table.)

expression[,...]

Specifies data to be written as a single record to the output file. You can specify data items using character string expressions, which may be symbol names, character strings in quotation marks, literal numeric values, or a lexical function. See Section 6.1 of the *VMS DCL Concepts Manual* for more information on string expressions.

You can specify a list of expressions separated by commas; the command interpreter concatenates the items into one record and writes the record to the output file.

The maximum size of any record that can be written is less than 1024 bytes. If, however, you specify the /SYMBOL qualifier, the maximum record size is 2048 bytes. (See the Description section.)

DESCRIPTION

In this example, the write command can write records to sequential, relative, or indexed sequential files that have been opened for writing. When the WRITE command writes a record, it always positions the record pointer after the record just written.

To write to a file, the file must be opened using either the /WRITE or the /APPEND qualifier with the OPEN command. However, the process-permanent files identified by the logical names SYS$INPUT, SYS$OUTPUT, SYS$ERROR, and SYS$COMMAND do not have to be explicitly opened to be written to.

If you do not specify the /SYMBOL qualifier, DCL places the command and the complete string expression (expanded if it was specified as one or more symbols) in a 1024-byte buffer. If you specify the /SYMBOL qualifier, DCL interprets the symbol or symbols and places the expanded string in a separate 2048-byte buffer, and then performs the write operation. For this reason, use the /SYMBOL qualifier where the record contains approximately 1000 bytes or more.

QUALIFIERS

/ERROR=label

Transfers control on an I/O error to the location specified by *label* (in a command procedure). If no error routine is specified and an error occurs during the writing of the file, the current ON condition action is taken. /ERROR overrides any ON condition action specified. If an error occurs and control passes successfully to the target label, the reserved global symbol $STATUS retains the error code.

/SYMBOL

Causes the expression to be interpreted and its expanded value placed in a 2048-byte (instead of a 1024-byte) buffer before the write operation is performed. If you specify multiple expressions, their values are concatenated and placed in the 2048-byte buffer. Use the /SYMBOL qualifier to write a very large record. (See the Description section.)

Each expression specified must be a symbol. You cannot specify character string expressions (that is, strings in quotation marks) with the /SYMBOL qualifier.

If you do not use the /SYMBOL qualifier, the entire command, including the expression or expressions, is placed in a 1024-byte buffer, as explained in the Description section.

/UPDATE

Replaces the last record read with the record specified with the expression parameter. You must be able to read and write to a file to use the /UPDATE qualifier. Use the WRITE/UPDATE command only after a READ command. The WRITE/UPDATE command modifies the last record you have read.

With sequential files, you must replace a record with another record of the same size when you use the WRITE/UPDATE command.

EXAMPLES

1
```
$ WRITE SYS$OUTPUT "Beginning second phase of tests"
```

The WRITE command writes a single line of text to the current output device.

2
```
$ OPEN/WRITE OUTPUT_FILE TESTFILE.DAT
$ INQUIRE ID "Assign Test-id Number"
$ WRITE/ERROR=WRITE_ERROR  OUTPUT_FILE  "Test-id is ",ID
$ WRITE/ERROR=WRITE_ERROR  OUTPUT_FILE  ""
$ !
$ WRITE_LOOP:
    .
    .
    .
$ GOTO WRITE_LOOP
$ END_LOOP:
$ !
$ CLOSE OUTPUT_FILE
$ PRINT TESTFILE.DAT
$ EXIT
$ !
$ WRITE_ERROR:
$ WRITE SYS$OUTPUT "There was a WRITE error."
$ CLOSE OUTPUT_FILE
$ EXIT
```

In this example, the open command opens the file TESTFILE.DAT; the INQUIRE command requests an identification number to be assigned to a particular run of the procedure. The number entered is equated to the symbol ID. The WRITE commands write a text line concatenated with the symbol name ID and a blank line.

The lines between the label WRITE_LOOP and END_LOOP process information and write additional data to the file. When the processing is finished, control is transferred to the label END_LOOP. The CLOSE and PRINT commands at this label close the output file and queue a copy of the file to the system printer.

The label WRITE_ERROR is used as the target of the /ERROR qualifier on the WRITE command; if an error occurs when a record is being written, control is transferred to the label WRITE_ERROR.

3
```
$ OPEN/APPEND OUTPUT_FILE TRNTO::DBA1:[PGM]PLAN.DAT
$ WRITE OUTPUT_FILE "BEGINNING PHASE 3"
```

In this example, the OPEN/APPEND command opens the file PLAN.DAT at the remote node TRNTO and positions the pointer at the end of the file. The WRITE command writes a record to the end of the file PLAN.DAT.

4
```
$ OPEN/APPEND MYFILE [JONES]TESTING.DAT
$ WRITE/SYMBOL MYFILE A,B,C
```

This example assumes that the symbols A, B, and C have already been defined. The OPEN/APPEND command opens the file [JONES]TESTING.DAT and positions the pointer at the end of the file. The WRITE/SYMBOL command concatenates the values of the symbols A, B, and C and writes this data to a new record at the end of the file.

Index

A

B

C

D

E

F

G

H

I

J

L

M

N

O

P

Q

R

S

T

U

V

W

Reader's Comments

VMS DCL Dictionary
AA–LA12A–TE

Please use this postage-paid form to comment on this manual. If you require a written reply to a software problem and are eligible to receive one under Software Performance Report (SPR) service, submit your comments on an SPR form.

Thank you for your assistance.

I rate this manual's:	Excellent	Good	Fair	Poor
Accuracy (software works as manual says)	☐	☐	☐	☐
Completeness (enough information)	☐	☐	☐	☐
Clarity (easy to understand)	☐	☐	☐	☐
Organization (structure of subject matter)	☐	☐	☐	☐
Figures (useful)	☐	☐	☐	☐
Examples (useful)	☐	☐	☐	☐
Index (ability to find topic)	☐	☐	☐	☐
Page layout (easy to find information)	☐	☐	☐	☐

I would like to see more/less _____

_____

_____

What I like best about this manual is _____

_____

_____

What I like least about this manual is _____

_____

_____

I found the following errors in this manual:

Page	Description
_____	_____
_____	_____
_____	_____
_____	_____
_____	_____

Additional comments or suggestions to improve this manual:

_____

_____

_____

_____

I am using **Version** _____ of the software this manual describes.

Name/Title _____ Dept. _____

Company _____ Date _____

Mailing Address _____

_____ Phone _____

— Do Not Tear - Fold Here and Tape ——————————————————————

digital™

No Postage
Necessary
if Mailed
in the
United States

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 33 MAYNARD MASS.

POSTAGE WILL BE PAID BY ADDRESSEE

DIGITAL EQUIPMENT CORPORATION
Corporate User Publications—Spit Brook
ZK01–3/J35 110 SPIT BROOK ROAD
NASHUA, NH 03062-9987

— Do Not Tear - Fold Here ——————————————————————

Cut Along Dotted Line

Reader's Comments

VMS DCL Dictionary
AA–LA12A–TE

Please use this postage-paid form to comment on this manual. If you require a written reply to a software problem and are eligible to receive one under Software Performance Report (SPR) service, submit your comments on an SPR form.

Thank you for your assistance.

I rate this manual's:	Excellent	Good	Fair	Poor
Accuracy (software works as manual says)	☐	☐	☐	☐
Completeness (enough information)	☐	☐	☐	☐
Clarity (easy to understand)	☐	☐	☐	☐
Organization (structure of subject matter)	☐	☐	☐	☐
Figures (useful)	☐	☐	☐	☐
Examples (useful)	☐	☐	☐	☐
Index (ability to find topic)	☐	☐	☐	☐
Page layout (easy to find information)	☐	☐	☐	☐

I would like to see more/less _____

_____

_____

What I like best about this manual is _____

_____

_____

What I like least about this manual is _____

_____

_____

I found the following errors in this manual:

Page	Description
_____	_____
_____	_____
_____	_____
_____	_____
_____	_____

Additional comments or suggestions to improve this manual:

_____

_____

_____

_____

I am using **Version** _____ of the software this manual describes.

Name/Title _____ Dept. _____

Company _____ Date _____

Mailing Address _____

_____ Phone _____

-— Do Not Tear - Fold Here and Tape ----------

digital™

No Postage
Necessary
if Mailed
in the
United States

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 33 MAYNARD MASS.

POSTAGE WILL BE PAID BY ADDRESSEE

DIGITAL EQUIPMENT CORPORATION
Corporate User Publications—Spit Brook
ZK01–3/J35 110 SPIT BROOK ROAD
NASHUA, NH 03062-9987

-— Do Not Tear - Fold Here ----------