digital

VAX/VMS Command Language User's Guide

Order No. AA-D023B-TE

VAX11

March 1980

The manual describes the VAX/VMS command language, DCL. It provides detailed reference information and examples of all nonprivileged commands available to general users.

VAX/VMS
Command Language User's Guide

Order No. AA-D023B-TE

SUPERSESSION/UPDATE INFORMATION:	This revised document supersedes the VAX/VMS Command Language User's Guide (Order No. AA-D023A-TE) and Update Notice No. 1 (AD-D023A-T1)
OPERATING SYSTEM AND VERSION:	VAX/VMS V02
SOFTWARE VERSION:	VAX/VMS V02

digital equipment corporation · maynard, massachusetts

First Printing, August 1978
Updated, February 1979
Revised, March 1980

DIGITAL	DECsystem-10	MASSBUS
DEC	DECtape	OMNIBUS
PDP	DIBOL	OS/8
DECUS	EDUSYSTEM	PHA
UNIBUS	FLIP CHIP	RSTS
COMPUTER LABS	FOCAL	RSX
COMTEX	INDAC	TYPESET-8
DDT	LAB-8	TYPESET-11
DECCOMM	DECSYSTEM-20	TMS-11
ASSIST-11	RTS-8	ITPS-10
VAX	VMS	SBI
DECnet	IAS	PDT
DATATRIEVE	TRAX	

CONTENTS

CONTENTS

CONTENTS

CONTENTS

CONTENTS

FIGURES

TABLES

PREFACE

MANUAL OBJECTIVES

This manual describes the VAX/VMS command language, DCL (DIGITAL Command Language), and provides usage and reference information on the command language.

INTENDED AUDIENCE

This manual is intended for all users of the VAX/VMS operating system, including applications programmers, system programmers, operators, and managers.

The VAX/VMS Primer is a tutorial manual that introduces the VAX/VMS operating system and the use of the command language. General users who are not familiar with interactive computer systems should read the Primer before using this command language user's guide.

The VAX/VMS Summary Description introduces operating system concepts of general interest. It is recommended that system programmers and managers be familiar with the material in the summary description before using this command language user's guide.

The VAX/VMS Guide to Using Command Procedures is a tutorial manual that defines and illustrates good practices in constructing command procedures with DCL commands. DCL users should read that guide to learn how to create effective command procedures using commands and lexical functions described in this user's guide.

STRUCTURE OF THIS DOCUMENT

The manual has three parts:

PART I. USING THE COMMAND LANGUAGE

This part is tutorial; it provides an overview of the command language and operating system concepts. Part I contains five chapters:

- Chapter 1, "Overview," describes how to access the system and enter commands. At the end of Chapter 1, on colored pages, is a table that summarizes all the DCL commands (including those for operators) in alphabetical order.

- Chapter 2, "File Specifications and Logical Names," explains the device and file naming conventions of the operating system and describes how to assign and use logical names to refer to devices and files in commands and programs.

- Chapter 3, "Disk and Tape Volumes," discusses concepts related to accessing files on disk and tape volumes, and provides examples of initializing and using disks and tapes.

- Chapter 4, "Programming with VAX/VMS," gives an overview of the program development tools provided by DCL commands and describes the environment in which programs execute.

- Chapter 5, "Grammar Rules," describes the syntax of the command language and defines the rules for entering commands, command parameters and qualifiers, numeric values and expressions, and character data.

PART II. COMMAND DESCRIPTIONS

This part contains detailed descriptions of each command. The commands are listed in alphabetical order, with the command name appearing at the top of the first and every page of the individual command description.

Readers of Part II should be familiar with the material covered in Part I. Furthermore, while the VAX/VMS Guide to Using Command Procedures is not a strict requirement for reading and using Part II, it is recommended. You may find it helps clarify some of the examples in Part II that involve command procedures.

Many of the DCL commands in Part II invoke language compilers that are available as separate products. The primary reference source for each of these commands is the user's guide published for the product.

APPENDIX

The appendix describes the foreign command feature of DCL.

ASSOCIATED DOCUMENTS

See the VAX-11 Information Directory and Index for a description of related documents and to obtain the order numbers of manuals referred to in this book.

CONVENTIONS USED IN THIS DOCUMENT

Convention	Meaning
(RET)	A symbol with a one- to three-character abbreviation indicates that you press a key on the terminal, for example, (RET) or (ESC) .
CTRL/x	The phrase CTRL/x indicates that you must press the key labeled CTRL while you simultaneously press another key, for example CTRL/C, CTRL/Y, CTRL/O. In examples, this control key sequence is shown as ^x, for example ^C, ^Y, ^O, because that is how the system echoes control key sequences.

Convention	**Meaning**
`$ SHOW TIME` `05-JUN-1980 11:55:22`	Command examples show all output lines or prompting characters that the system prints or displays in black letters. All user-entered commands are shown in red letters.
`$ FORTRAN MYFILE` `$ LINK MYFILE` `$ RUN MYFILE`	Examples showing the contents of files are enclosed in boxes.
`$ TYPE MYFILE.DAT` . . .	Vertical series of periods, or ellipsis, mean that not all the data that the system would display in response to the particular command is shown; or, that not all the data a user would enter is shown.
file-spec,...	Horizontal ellipsis indicates that additional parameters, values, or information can be entered.
[logical-name]	Square brackets indicate that the enclosed item is optional. (Square 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.)
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.

SUMMARY OF TECHNICAL CHANGES

This manual introduces many new commands as well as new qualifiers for existing commands. Other software changes documented in this manual are listed below as "new features".

NEW COMMANDS

The alphabetized list below identifies the new commands introduced in this manual. Note that the DIRECTORY command is included in this list to emphasize that it has been redesigned and is effectively new.

New Commands

ANALYZE	MAIL	SET PASSWORD
BASIC	MESSAGE	SHOW QUOTA
CORAL	PASCAL	STOP/ABORT
DIRECTORY	PATCH	STOP/ENTRY
EDIT/EDT	SET HOST	STOP/REQUEUE
EDIT/SUM	SET MESSAGE	

NEW QUALIFIERS

The new qualifiers identify software enhancements you may want to begin using at once. They are:

Command	**New Qualifier**
COPY	/VOLUME
CREATE	/[NO]LOG /VOLUME
CREATE/DIRECTORY	/[NO]LOG /VERSION_LIMIT /VOLUME
DUMP	/FILE_HEADER /[NO]FORMATTED

Command	New Qualifier
EDIT/SLP	/[NO]CHECKSUM /[NO]REPORT
EDIT/SOS	/[NO]BAK /[NO]NUMBERS
FORTRAN	/[NO]F77 /[NO]G_FLOATING
INQUIRE	/[NO]PUNCTUATION
JOB	/CPUTIME /WSDEFAULT /WSQUOTA
HELP	/LIBRARY
LIBRARY	/CROSS_REFERENCE /HELP /[NO]LOG /MODULE /TEXT /WIDTH
LINK	/HEADER /P0IMAGE /PROTECT /[NO]USERLIBRARY
MACRO	/UPDATE
MOUNT	/[NO]CACHE /[NO]HDR3 /[NO]QUOTA
PRINT	/CHARACTERISTICS
RENAME	/[NO]CONFIRM /[NO]LOG
SET PROCESS	/PRIVILEGES
SET PROTECTION	/[NO]CONFIRM /[NO]LOG
SET QUEUE/ENTRY	/CHARACTERISTICS /CPUTIME /WSDEFAULT /WSQUOTA
SET TERMINAL	/[NO]FORM /FT1 /FT2 /FT3 /FT4 /FT5 /FT6 /FT7 /FT8 /[NO]FULLDUP /[NO]HALFDUP /LA120 /VT100

Command	**New Qualifier**
SHOW SYSTEM	/BATCH /NETWORK /PROCESS /SUBPROCESS
SUBMIT	/CPUTIME /WSDEFAULT /WSQUOTA
UNLOCK	/[NO]CONFIRM /[NO]LOG

QUALIFIERS REMOVED

Three qualifiers have been removed. The /LOCAL and /REMOTE qualifiers have been removed from the SET TERMINAL command, and the /WORK_FILES qualifier has been removed from the FORTRAN command.

NEW FEATURES

Other major modifications to this manual reflect the following software changes:

- Addition of the user privileges BYPASS, PFNMAP, SHMEM, and SYSPRV (see Table 1-3)
- Addition of wild card characters (see Section 2.1.6)
- Addition of device codes to accommodate new hardware (see Table 2-1)
- Additions of new default file types (see Table 2-2)
- Modifications to the displays in the examples for the DIRECTORY, SHOW NETWORK, SHOW QUEUE, and SHOW SYSTEM commands (see Part II)
- Introduction of pooled quotas (see changes in the RUN Process command, Part II)
- Introduction of disk quotas and improvements in multivolume disk handling (see Chapter 3)

ORGANIZATIONAL CHANGES TO THE MANUAL

Some of the command descriptions in Part II are now presented in two distinct parts. For example, the MACRO command now appears as MACRO and MACRO/RSX11. The commands in this group are CREATE, EDIT, LIBRARY, MACRO, SET PROCESS, and SET PROTECTION.

The former Chapter 5, "Command Procedures and Batch Jobs," has been removed. A new manual, the VAX/VMS Guide to Using Command Procedures, expands this subject.

A new appendix entitled "Foreign Command Feature of DCL" is introduced as Appendix A.

Numerous other small changes have been made to incorporate corrections, additions, clarifications, and minor formatting or syntax improvements.

PART I

USING THE COMMAND LANGUAGE

CHAPTER 1

OVERVIEW

The DCL command language provides VAX/VMS users with an extensive set of commands for:

- Interactive program development
- Device and data file manipulation
- Interactive and batch program execution and control

The following sections show how to access the system and describe how the system processes commands interactively. These sections are followed by tables, on colored pages, that describe all the commands available to general users. The commands are listed in functional categories to provide a quick overview of VAX/VMS capabilities.

Additional DCL commands exist for users who have the operator user privilege (OPER). Minimal references to those commands are made in this manual; instead, you can find full descriptions of them in the VAX/VMS Operator's Guide.

1.1 ACCESSING THE SYSTEM

When a terminal is physically connected to the system, you can get its attention and signal that you want to begin a terminal session by pressing the RETURN or CTRL/Y key. The system responds by prompting you for your user name. After you enter your user name, the system prompts you to enter your password. For example:

```
RET
Username: PATTI RET
Password:
```

When you enter the password, the system does not echo it, that is, the system accepts the input line but does not display it on the terminal.

During this sequence, known as login, the system validates that you are authorized to use the system. As part of the login sequence, one or more command files may be executed. For example, the command files may display messages telling you about the system. Finally, the system indicates that it is ready to accept commands by displaying the prompting character, a dollar sign ($), as shown below:

```
        Welcome to VAX/VMS Version 2.00
              .
              .
              .
$
```

1.2 YOUR COMMAND ENVIRONMENT

When you have logged into the system and the system is ready to accept commands, the system defines the environment within which it responds to your requests. This environment is called a process. Associated with your process are various default characteristics, among which are the following:

- An account number, which your installation uses to keep track of the computer resources used.
- A user identification code (UIC), which provides you with a group number and a member number within the group. Other users can have the same group number. Within groups, users are allowed to share files or system resources more freely.
- A default disk device and directory name, which the system uses to locate and catalog files that you create or use.
- Default devices for the input, output, and error streams of the process, from which the system reads command and program input and to which the system writes messages.
- A set of privileges and resource quotas that define what system resources or system functions you, or programs you execute, will be allowed to use. Tables 1-3 and 1-4 at the end of this chapter summarize the specific privileges and quotas the system defines.
- A command interpreter -- the operating system component responsible for reading and translating your typed-in requests. This manual describes only the DCL command interpreter.

The system obtains the characteristics unique to your process from the user authorization file. The user authorization file is a list of all users who can access the system; the system manager or operator maintains this file.

1.3 ENTERING COMMANDS

Commands consist of English-language words (generally verbs) that describe what you want the system to do. Commands can optionally contain qualifiers and parameters. Command qualifiers modify a command. They provide the system with additional information on how to execute the command. Command parameters describe the object of the command. In some cases, a parameter is a keyword; in other cases, it is a file or a device to manipulate or a program to execute.

The following example shows a PRINT command and the system's response, as they would appear on a terminal:

```
$ PRINT/COPIES=2 MYFILE.DAT              ! Print the file RET
  Job 210 entered on queue LPB0:
$
```

The elements of the example above are analyzed below:

$	The system prompt for command input; a dollar sign means that the command interpreter is ready for you to type a command.
PRINT	The command name, requesting the system to queue a file for printing on the system printer.
/COPIES=2	A qualifier to the PRINT command, requesting that two copies of the specified file be printed. A qualifier is always preceded with a slash character (/), and, if it requires a value (as in this example), the qualifier name is separated from the value with either an equal sign (=) or a colon (:).
MYFILE.DAT	The command parameter, naming the file to be printed. In this command, as in many DCL commands, the parameter is a file specification. At least one blank space must always immediately precede a parameter.
! Print the file	A comment. You can use comments, as needed, to document terminal sessions or command procedures.
RET	The carriage return. Pressing this key after entering a full command line terminates the command input; the system begins processing the command. Note that examples in this manual do not explicitly show the RET following commands; you can assume that all lines shown in examples must be terminated with RET unless stated otherwise.
Job 210 entered on queue LPB0:	A message from the PRINT command, indicating that the command completed successfully; the command interpreter gave the print job an identification number and queued it to the queue named LPB0.
$	The next $ prompt, indicating that the PRINT command has completed successfully; that is, two copies of the file MYFILE.DAT were queued for printing and the system is ready to accept another command.

Chapter 5, "Grammar Rules," contains complete details on the syntax rules for entering commands, parameters, and qualifiers, including:

- Continuing commands on more than one line
- Shortening command and other keyword names to four or fewer characters
- Specifying values for qualifiers

1.4 COMMAND PROMPTING

When you enter a command at the terminal, you need not enter the entire command all at once. If you enter a command that requires parameters and do not specify any parameters, the command interpreter prompts you for all remaining parameters, including optional parameters. For example:

```
$ PRINT/COPIES=2
$_File:      MYFILE.DAT
 Job 211 entered on queue LPA1
```

In this example, no parameter is entered, so the system prompts for a file specification. A line beginning with $_ indicates that the system is in prompt mode.

When you are prompted for an optional parameter, you can press <RET> to complete the command sequence without specifying the parameter, as the following example illustrates:

```
$ ALLOCATE
$_Device:   DM:
$_Log_Name:<RET>
 _DMB1: ALLOCATED
```

In the above example, the ALLOCATE command is entered with no parameters; the command interpreter begins prompting for all parameters. The second prompt, $_Log_Name:, is for an optional logical name parameter. A null entry, signaled by <RET>, terminates the command entry.

1.5 SYSTEM MESSAGES

When you enter a command incorrectly, the command interpreter issues a descriptive error message telling you what was wrong. For example, if you specify more than one parameter for a command that accepts a single parameter, you receive the message:

```
%DCL-W-MAXPARM, maximum parameter count exceeded
```

You must then retype the command correctly.

Other error messages may occur during execution of a command. These messages can indicate errors such as a nonexistent file or a conflict in qualifiers.

Not all messages from the system indicate errors; some messages merely warn you of a particular condition.

Messages begin with a percent sign (%) or hyphen (-) and have the general format:

```
%FACILITY-L-IDENT, text
```

where FACILITY is a mnemonic for the operating system facility or the program issuing the message, L is a severity level indicator (S for success, I for informational, W for warning, E for error, and F for fatal), and IDENT is a shorthand code for the message text. When a series of messages is issued, messages after the first one are prefixed by a hyphen rather than a percent sign. Note that it is possible to change the error message format display with the SET MESSAGE command.

Because the messages are descriptive, you can usually tell what you need to do when you issue the command again.

The VAX/VMS System Messages and Recovery Procedures Manual lists the system messages and describes what you can do to correct an error.

1.6 THE HELP COMMAND

You may not always have this user's guide available at your terminal when you need a summary of the format of a particular command or a list of its valid qualifiers. The HELP command provides you with this information. For example, you might type the command:

```
$ HELP PRINT
```

The system responds by displaying an abstract of the PRINT command and keywords you can enter as parameters to the HELP command to obtain more information about PRINT. If you enter the following:

```
$ HELP PRINT QUALIFIERS
```

The HELP command displays a description of each PRINT command qualifier.

1.7 TAILORING THE COMMAND LANGUAGE

Beyond the normal syntax and predefined command names and parameters, VAX/VMS lets you define synonyms for command names and lets you "create" commands by writing command procedures.

1.7.1 Synonyms for DCL Commands

You can create synonyms by using symbolic names. You define symbolic names with assignment statements that equate a symbol name to a numeric value or a character string. For example, you could define a symbol name to be equated to a real DCL command name as follows:

```
$ LIST := DIRECTORY
```

For the remainder of the terminal session, when you enter the symbol LIST as the first word on a command line, the command interpreter substitutes the symbol with the string DIRECTORY and executes the DIRECTORY command.

A symbol can also contain a portion of a command, for example:

```
$ PDEL := DELETE SYS$PRINT/ENTRY=
```

This assignment statement equates the symbol name PDEL with the command necessary to delete an entry from the printer queue, except that the required value for the /ENTRY qualifier is omitted. When you use this symbol as a command synonym, you must also type the job identification number assigned to the entry, as shown below:

```
$ PDEL 210
```

When the command interpreter processes this line, it substitutes the symbol PDEL with its current value and then executes the command:

```
     DELETE SYS$PRINT/ENTRY=210
```

For additional information on defining symbols for DCL command synonyms, see Section 5.11, the description of the = (Assignment Statement) command in Part II, and Appendix A.

1.7.2 **Command Procedures**

A command procedure is a file that contains a sequence of DCL commands, some of which can conditionally control the execution of the procedure. By placing sets of frequently used commands and/or qualifiers in command procedures, you can construct command language "programs" from DCL commands. After you create a command procedure, you can execute all the commands in it with a single command. For example, suppose a procedure named TESTALL.COM contains the command lines:

```
$ RUN WEEKCALC
$ RUN TIMECARDS
$ PRINT WEEKCALC.OUT,TIMECARDS.OUT
```

You can execute the three commands in this file by entering:

```
     $ @TESTALL
```

Another way to execute these commands is to enter the SUBMIT command, which requests the system to process a command procedure as a batch job. This way your terminal remains free for interactive work.

The VAX/VMS Guide to Using Command Procedures provides details on using the command interpreter's symbolic capabilities and on developing and using command procedures. Reference information on each of these commands can be found in Part II.

1.8 **THE FILE SYSTEM**

Using DCL commands, you can create, access, and update data files and programs. The VAX-11 Record Management Services (RMS) provide the access and control capabilities that are called by the DCL commands you request.

You can also define and access files from within your programs by using RMS or by using the input/output services of the VAX/VMS operating system directly.

Chapter 2, "File Specifications and Logical Names," describes how to identify and refer to files. It also describes the directory structure of the files on disk volumes and explains how to create a hierarchy of directories to catalog and maintain files.

Each of the commands for file manipulation is described in detail in Part II of this manual. For a list of additional manuals that contain information on file services and utilities, see the VAX-11 Information Directory and Index.

1.9 TERMINAL CHARACTERISTICS

Your terminal keyboard is the means by which you communicate with the command interpreter or enter data solicited by a command or program. Although a terminal keyboard is similar to that of a typewriter, there are several differences between the way a typewriter and your terminal accept and display data that you type. These differences include:

- The interpretation of uppercase and lowercase alphabetic characters
- The type-ahead buffer
- Special function keys

When you type input lines to the command interpreter from a terminal that displays lowercase letters, the command interpreter translates all lowercase characters to uppercase before it executes a command, unless you enter character strings enclosed in quotation marks.

After you enter a command and while the command interpreter is responding to it or executing it, your keyboard is not locked; that is, you can continue typing. The system saves what you type during this time in a special buffer called the type-ahead buffer. It does not, however, echo what you type. When the command currently being executed completes, the command interpreter displays what you have typed.

If you typed a full command line, including a RET , the command interpreter executes the command after displaying it. Otherwise, it displays what you have typed and waits for you to enter the rest of the line. If you have entered more than one complete line, it displays each line just before executing it.

Some of the special terminal function keys provide line-editing functions so you can correct or cancel what you type before passing it to the command interpreter. Other keys can interrupt system processing. Some of the line-editing keys are described in the next section. Table 1-1 lists all the terminal function keys and describes their use.

1.9.1 Special Terminal Function Keys

As you type commands or program data at the terminal, you can take advantage of several keys that let you make changes to lines as you type them. Thus, if you make a mistake, you can correct it before pressing the return key.

1.9.1.1 Deleting Characters

1.9.1.1 Deleting Characters - The DEL (DELETE) key backspaces over the most recently entered character and deletes it. For example:

```
$ PRO DEL INT
```

After incorrectly typing the letter "O", you can delete it by pressing DEL and then continue your input by typing "INT". On a hardcopy terminal, the letters deleted are displayed between backslash characters so you can see what is being deleted. For example:

```
$ PRO\O\INT
```

On a video display terminal, pressing DEL actually erases the character from the screen and moves the cursor backwards. Note that the key that performs the delete function is marked RUBOUT on some terminals.

1.9.1.2 **Deleting Lines** - To cancel an entire line, you can use the CTRL/U key sequence. To use the CTRL key, you must press it at the same time you press the other letter (in this case, U). For example, if you make several mistakes on a particular line and want to cancel the line and reenter it, use CTRL/U as shown below:

```
     $ PRNT/CIPY=2^U
     $ PRINT/COPIES=2 MYFILE.DAT
```

When you cancel a line with CTRL/U, the system ignores the line and prompts you for the next line.

1.9.1.3 **Canceling Commands** - If you are entering commands in response to prompts and want to discard the entire command you have entered so far (not just the most recently entered line), use CTRL/C or CTRL/Y, as shown below:

```
     $ PRINT/COPIES=3
     $_File:     MYFILE.DAT^Y
```

In this example, the PRINT command was entered correctly, and the system prompted for the name of a file to print. However, while entering the file name, the user decided not to enter the command at all, and used CTRL/Y to discard the entire command.

1.9.2 Setting Terminal Characteristics

There are many types of terminals, and each has its own operating characteristics. You can change some of these characteristics, based on your requirements, with the SET TERMINAL command. To determine the current characteristics of your terminal, issue the SHOW TERMINAL command, as shown below:

```
     $ SHOW TERMINAL
       TTF3: /VT52, WIDTH=80, PAGE=24, OWNER=SELF
             SPEED=(2400,2400), CRFILL=0, LFFILL=0, NO PARITY
             INTERACTIVE, ECHO, TYPEAHEAD, NOESCAPE, NOHOSTSYNC,
             TTSYNC, UPPERCASE, TAB, WRAP, SCOPE, LOCAL, NOHOLDSCREEN,
             NOEIGHTBIT, BROADCAST, NOREADSYNC, NOFORM, HALFDUP,
```

Most of the parameters displayed by the SHOW TERMINAL command can be changed by corresponding qualifiers for the SET TERMINAL command. For example, you can change the case of the letters from uppercase to lowercase with this command:

```
     $ SET TERMINAL/LOWERCASE
```

After you issue this command, all lowercase alphabetic characters are displayed in lowercase (if your terminal is capable of displaying lowercase letters).

Table 1-1
Terminal Function Keys

Key	Function
RETURN	Carriage return; transmits the current line to the system for processing. (On some terminals, the RETURN key is labeled CR.) Before a terminal session, initiates login sequence.
Control keys	Define functions to be performed when the CTRL key and another key are pressed simultaneously. All CTRL/x key sequences are echoed on the terminal as ^x.
CTRL/C and CTRL/Y 1	During command entry, cancels command processing. Before a terminal session, initiates login sequence. Interrupts command or program execution and returns control to the command interpreter.
CTRL/I	Duplicates the function of the TAB key.
CTRL/K	Advances the current line to the next vertical tab stop.
CTRL/L	Form feed.
CTRL/O	Alternately suppresses and continues display of output to the terminal.
CTRL/Q	Restarts terminal output that was suspended by CTRL/S.
CTRL/R	Retypes the current input line and leaves the cursor positioned at the end of the line.
CTRL/S	Suspends terminal output until CTRL/Q is pressed.
CTRL/U	Discards the current input line.
CTRL/X	Discards the current line and deletes data in the type-ahead buffer.
CTRL/Y	(See CTRL/C.)
CTRL/Z	Signals end-of-file for data entered from the terminal.

1. Certain system and user programs provide special routines to respond to CTRL/C interrupts. If CTRL/C is pressed to interrupt a program that does not handle CTRL/C, CTRL/C has the same effect as CTRL/Y and echoes as ^Y.

(continued on next page)

Table 1-1 (Cont.)
Terminal Function Keys

Key	Function
DELETE	Deletes the last character entered at the terminal and backspaces over it. (On some terminals, the DELETE key is labeled RUBOUT.)
ESCAPE	Has special uses to particular commands or programs, but generally performs the same function as RETURN. (On some terminals, the ESCAPE key is labeled ALTMODE or ESC (SEL).)
TAB	Moves the printing element or cursor on the terminal to the next tab stop on the terminal. The system provides tab stops at every 8 character positions on a line.

1.10 SUMMARY OF VAX/VMS DCL COMMANDS

Table 1-2 summarizes the VAX/VMS DCL commands in alphabetical order. A brief description of the function performed by each command is included.

To provide more of a system overview, the table includes all the DCL commands, including those described in the VAX/VMS Operator's Guide. The operator commands are identified by a superscript 1.

Table 1-2
Summary of VAX/VMS DCL Commands

Command	Function
Assignment statements	Assign strings as synonyms for all or a portion of a DCL command
=	Arithmetic assignment - equates a local symbol name to an arithmetic expression or constant
==	Arithmetic assignment - equates a global symbol name to an arithmetic expression or constant
:=	String assignment - defines a local symbol name as a synonym for all or a portion of a DCL command
:==	String assignment - defines a global symbol name as a synonym for all or a portion of a DCL command

(continued on next page)

Table 1-2 (Cont.)
Summary of VAX/VMS DCL Commands

Command	Function
@ (execute procedure)	Executes a command procedure or places data from a command file into the input stream
ALLOCATE	Reserves a device for use by a single user and, optionally, assigns a logical name to the device
ANALYZE	Describes the contents of an object file or the symbol information appended to a shareable image file
APPEND	Adds the contents of one or more files to the end of another file
ASSIGN	Defines a file specification or a device name to be associated with a logical name for subsequent use in commands or programs
ASSIGN/MERGE[1]	Removes the jobs from one queue and places them in another queue
ASSIGN/QUEUE[1]	Assigns a logical queue to a device
BASIC	Invokes the VAX-11 BASIC compiler to enter and compile BASIC language source statements
BASIC/RSX11	Invokes the PDP-11 BASIC-PLUS-2 compiler to begin a BASIC session
BLISS	Invokes the VAX-11 BLISS-32 compiler to compile one or more BLISS-32 or common BLISS source programs
CANCEL	Halts periodic execution of an image scheduled for execution in a process
CLOSE	Cancels an input or output path to a sequential file or device
COBOL/C74	Invokes the VAX-11 COBOL-74 compiler to compile COBOL language source statements
COBOL/RSX11	Invokes the PDP-11 COBOL-74/VAX compiler to compile COBOL language source statements
CONTINUE	Resumes execution of an interrupted command, program, or command procedure

1. This DCL command is described in the VAX/VMS Operator's Guide.

(continued on next page)

Table 1-2 (Cont.)
Summary of VAX/VMS DCL Commands

Command	Function
COPY	Copies one or more files to one or more additional files
CORAL	Invokes the VAX-11 CORAL 66 compiler to compile one or more CORAL 66 source language programs
CREATE	Creates a file from data entered at the terminal or in the input stream
CREATE/DIRECTORY	Defines a new directory or subdirectory for cataloging files
DEALLOCATE	Relinquishes use of a previously allocated device, thus making the device available to other users
DEASSIGN	Cancels a logical name assignment made with the ALLOCATE, ASSIGN, or DEFINE command
DEASSIGN/QUEUE [1]	Deassigns a queue from a device
DEBUG	Invokes the VAX-11 Symbolic Debugger to begin or continue interactive debugging
DECK	Marks the beginning of records to be read as the input data stream for a command (required only when data contains a dollar sign ($) in the first position of any record)
DEFINE	Equates character strings with file specifications or logical names
DELETE	Removes a directory entry for a file or files and makes any data in the file(s) inaccessible
DELETE/ENTRY	Deletes an entry from a printer or batch job queue or stops processing of the current job
DELETE/QUEUE [1]	Deletes batch queues and printer queues
DELETE/SYMBOL	Deletes one or more symbol names from the local or global symbol tables for the process

1. This DCL command is described in the VAX/VMS Operator's Guide.

(continued on next page)

Table 1-2 (Cont.)
Summary of VAX/VMS DCL Commands

Command	Function
DEPOSIT	Replaces the contents of a location in virtual memory with new data or instructions
DIFFERENCES	Compares the contents of files and reports the differences between them
DIRECTORY	Displays information about a file or a group of files
DISMOUNT	Releases the connection between a user and a disk or tape volume that is currently mounted on a device
DUMP	Displays or prints the contents of a file or volume in ASCII, hexadecimal, octal or decimal format
EDIT/EDT	Begins an interactive editing session with the EDT editor to create or modify a file
EDIT/SLP	Provides input to the batch editor, SLP
EDIT/SOS	Begins an interactive editing session with the SOS editor to create or modify a file
EDIT/SUM	Invokes the SUMSLP batch-oriented editor to update a single input file with multiple files of edit commands
EOD	Marks the end of an input data stream begun with the DECK command
EOJ	Signals the end of a batch job submitted through a card reader
EXAMINE	Displays the contents of a location in virtual memory
EXIT	Terminates an image or command procedure processing at the current level
FORTRAN	Invokes the VAX-11 FORTRAN compiler to compile FORTRAN language source statements
GOTO	Transfers control to another statement in a command procedure

(continued on next page)

Table 1-2 (Cont.)
Summary of VAX/VMS DCL Commands

Command	Function
HELP	Displays information on the current output stream device from the system HELP files or any help library you specify
IF ... THEN	Compares expressions consisting of symbolic or literal values, or command or program status values, and performs a stated action based on the result of the test
INITIALIZE	Deletes all existing data, if any, on a mass storage volume, writes a label on the volume, and readies the volume for new data
INITIALIZE/QUEUE[1]	Creates batch queues and output queues
INQUIRE	Requests interactive assignment of a variable value for a symbol name
JOB	Marks the beginning of a batch job submitted through a card reader
Lexical Functions	Alternate representations for symbols or expressions that return information about character strings and attributes of the current process
LIBRARY	Creates or modifies libraries of various kinds
LIBRARY/RSX11	Creates or modifies RSX-11M macro libraries or object module libraries
LINK	Binds one or more object modules into an executable or shareable program image
LINK/RSX11	Invokes the RSX-11M Task Builder to link one or more object modules into an RSX-11M task
Login Procedure	Initiates communication between a user and the system
LOGOUT	Terminates communication between a user and the system
MACRO	Invokes the VAX-11 MACRO assembler to assemble a VAX-11 assembly language program

1. This DCL command is described in the VAX/VMS Operator's Guide.

(continued on next page)

Table 1-2 (Cont.)
Summary of VAX/VMS DCL Commands

Command	Function
MACRO/RSX11	Invokes the MACRO-11 assembler to assemble a PDP-11 assembly language program
MAIL	Invokes the Personal Mail Utility to send messages to other users of the system
MCR	Passes a command line to the RSX-11M Application Migration Executive, or places the terminal in MCR command mode
MCR BAD [1]	See RUN SYS$SYSTEM:BAD
MCR DSC1 [1]	See RUN SYS$SYSTEM:DSC1
MCR DSC2 [1]	See RUN SYS$SYSTEM:DSC2
MCR VFY1 [1]	See RUN SYS$SYSTEM:VFY1
MCR VFY2 [1]	See RUN SYS$SYSTEM:VFY2
MESSAGE	Invokes the Message Utility to compile one or more files of message definitions
MOUNT	Makes a disk or tape volume available for the reading and writing of files and, optionally, assigns a logical name to the device on which the volume is mounted
ON ... THEN	Defines the action to be taken when a command or program incurs errors of particular severity levels, or when the CTRL/Y function key is used
OPEN	Establishes a path to a file or a device for input or output operations
PASCAL	Invokes the VAX-11 PASCAL compiler to compile one or more PASCAL source programs
PASSWORD	Provides a password associated with a job entered through a card reader
PATCH	Invokes the VAX-11 PATCH Utility to patch an executable image, shareable image, or device driver image

1. This DCL command is described in the VAX-11 Utilities Reference Manual.

(continued on next page)

Table 1-2 (Cont.)
Summary of VAX/VMS DCL Commands

Command	Function
PRINT	Queues a file for printing or on a specified device
PURGE	Deletes old versions of a specified file or files
READ	Reads the next record from a sequential file or device and assigns the contents of the record to a symbol name
RENAME	Changes the name of a file or a group of files
REPLY[1]	Allows the operator to communicate with system users, selectively enable and disable operator status, and examine the log file
REQUEST	Displays a message at an operator's terminal
RUN (Image)	Places an executable image in execution in the current process
RUN (Process)	Creates a separate process to execute a specified image
RUN SYS$SYSTEM:BAD [2]	Locates and counts the bad blocks contained on Files-11 disks
RUN SYS$SYSTEM:DSC1 [2]	Transfers files contained on Files-11 Structure Level 1 disks to tapes or disks for back-up and storage
RUN SYS$SYSTEM:DSC2 [2]	Transfers files contained on Files-11 Structure Level 2 disks to tapes or disks for back-up and storage
RUN SYS$SYSTEM:INSTALL [3]	Installs or deletes known images
RUN SYS$SYSTEM:SYE [3]	Creates an error log report from a binary formatted file

1. This DCL command is described in the VAX/VMS Operator's Guide.

2. This DCL command is described in the VAX-11 Utilities Reference Manual.

3. This DCL command is described in the VAX/VMS System Manager's Guide.

(continued on next page)

Table 1-2 (Cont.)
Summary of VAX/VMS DCL Commands

Command	Function
RUN SYS$SYSTEM:VFY1 [1]	Checks the readability and validity of Files-11 Structure Level 1 disks
RUN SYS$SYSTEM:VFY2 [1]	Checks the readability and validity of Files-11 Structure Level 2 disks
SET ACCOUNTING [2]	Selectively enables or disables the recording of particular kinds of accounting information
SET CARD_READER	Defines the translation mode for a card reader
SET CONTROL_Y	Enables the use of the CTRL/Y function key to interrupt command execution
SET DEFAULT	Changes the directory and/or disk device used by default to locate and catalog files
SET DEVICE [1]	Establishes the spooling and error logging status on a specified device
SET HOST	Establishes a virtual communication link between a terminal and a network node to which the terminal is not directly connected
SET LOGINS [1]	Establishes the maximum number of users able to log into the system
SET MAGTAPE	Defines the density of a magnetic tape device or rewinds a tape
SET MESSAGE	Overrides or supplements the system messages
SET NOCONTROL_Y	Disables the use of the CTRL/Y function key to interrupt control execution
SET NOON	Disables previously declared ON conditions, thus preventing the command interpreter from taking any action on errors issued by command processing

1. This DCL command is described in the <u>VAX-11 Utilities Reference Manual</u>.

2. This DCL command is described in the <u>VAX/VMS Operator's Guide</u>.

(continued on next page)

Table 1-2 (Cont.)
Summary of VAX/VMS DCL Commands

Command	Function
SET NOVERIFY	Suppresses display of command lines in subsequently executed command procedures
SET ON	Restores command interpreter error actions in a command procedures
SET PASSWORD	Allows users to change their own passwords
SET PRINTER[1]	Establishes the characteristics of a specified line printer
SET PROCESS	Changes execution characteristics of a process
SET PROCESS/PRIORITY	Changes the base priority for a process
SET PROTECTION	Changes the protection applied to a file or a group of files, restricting or allowing access to the file by different categories of user
SET PROTECTION/DEFAULT	Establishes the default protection for all files subsequently created during the terminal session or batch job
SET PROTECTION/DEVICE[1]	Establishes the protection for a nonfile-structured device
SET QUEUE/ENTRY	Changes the current status or attributes of a file that is queued for printing or for batch job execution, but not yet processed by the system
SET RMS_DEFAULT	Defines default multiblock and multibuffer counts for VAX-11 RMS file operations
SET TERMINAL	Defines the characteristics of the terminal
SET TIME[1]	Resets the system clock to the specified value
SET UIC[1]	Establishes a new user identification code as the process UIC

1. This DCL command is described in the VAX/VMS Operator's Guide.

(continued on next page)

Table 1-2 (Cont.)
Summary of VAX/VMS DCL Commands

Command	Function
SET VERIFY	Causes all command lines in command procedures subsequently executed to be displayed at the terminal or printed in the batch job log file
SET WORKING_SET	Establishes a default working set size for images executed in the current process
SHOW DAYTIME	Displays the current date and time of day on the current output device
SHOW DEFAULT	Displays the current default directory and disk device
SHOW DEVICES	Displays the status of devices in the system
SHOW LOGICAL	Displays the current assignments of logical names and equivalence names made by the ASSIGN, ALLOCATE, DEFINE, or MOUNT commands
SHOW MAGTAPE	Displays characteristics of a magnetic tape device
SHOW NETWORK	Displays the availability of the local node as a member of the network and the names of all nodes currently accessible by the local node
SHOW PRINTER	Displays the characteristics of a line printer
SHOW PROCESS	Displays information about the current process, including subprocesses, privileges, quotas, and accounting information
SHOW PROTECTION	Displays the default protection applied to new files created
SHOW QUEUE	Displays the names, job numbers, and status of current and pending jobs in the printer and batch job queues
SHOW QUOTA	Displays the current disk quota that is authorized and used by a specific user on a specific disk
SHOW RMS_DEFAULT	Displays the current multiblock and multibuffer counts for VAX-11 RMS operations

(continued on next page)

Table 1-2 (Cont.)
Summary of VAX/VMS DCL Commands

Command	Function
SHOW STATUS	Displays information about the image currently executing in the process
SHOW SYMBOL	Displays current local or global symbols and the strings or values assigned to them
SHOW SYSTEM	Displays the current status of processes in the system
SHOW TERMINAL	Displays the current characteristics of the terminal
SHOW TIME	Displays the current date and time on the current output device
SHOW TRANSLATION	Searches all three logical name tables for a logical name and displays the equivalence name of the first match found
SHOW WORKING_SET	Displays the current working set default and limits
SORT	Invokes the VAX-11 SORT Utility to sort the records in a file based on one or more key fields within each record
SORT/RSX11	Invokes the PDP-11 SORT Utility to sort the records in a file based on one or more key fields within each record
START/QUEUE[1]	Starts batch queues and printer queues
STOP	Halts execution of a command procedure, program, or a subprocess or detached process
STOP/ABORT	Stops the printing of a job that is currently being printed
STOP/ENTRY	Stops the execution of a batch job that is currently running
STOP/QUEUE[1]	Suspends batch queues and printer queues
STOP/REQUEUE	Stops the printing of a job that is currently being printed and requeues that job at the end of the queue

1. This DCL command is described in the VAX/VMS Operator's Guide.

(continued on next page)

Table 1-2 (Cont.)
Summary of VAX/VMS DCL Commands

Command	Function
SUBMIT	Enters one or more command procedures in a batch job queue
SYNCHRONIZE	Places the process issuing this command in a wait state until a specified batch job completes
TYPE	Displays the contents of a file or files on the current output device
UNLOCK	Allows access to a file that was not properly closed
WAIT	Places the current process in a wait state for a specified period of time
WRITE	Writes a single record consisting of one or more character strings or evaluated symbols to a sequential file or device

1.11 SUMMARY OF VAX/VMS USER PRIVILEGES AND RESOURCE QUOTAS

Tables 1-3 and 1-4 summarize the full set of user privileges and resource quotas. The system manager establishes default values for these that normally are established at login time for your terminal session. Section 4.4.1 discusses quotas and privileges in general terms. Where special quotas or privileges affect the use of DCL commands, the command descriptions in Part II note them.

Table 1-3
User Privileges

Name	Privilege
ACNT	Create a process for which no accounting records are made
ALLSPOOL	Allocate spooled devices
ALTPRI	Increase the base execution priority for any process
BUGCHK	Make bug check error log entries
BYPASS	Bypass UIC protection
CMEXEC	Change mode to executive
CMKRNL	Change mode to kernel

(continued on next page)

Table 1-3 (Cont.)
User Privileges

Name	Privilege
DETACH	Create detached processes
DIAGNOSE	Issue diagnostic I/O requests
EXQUOTA	Exceed resource quotas
GROUP	Control execution of other processes in the same group
GRPNAM	Enter names in the group logical name table
LOG_IO	Perform logical I/O functions
MOUNT	Execute a mount volume I/O function
NETMBX	Create a network device
OPER	Perform operator functions
PFNMAP	Create or delete sections mapped by page frame number
PHY_IO	Perform physical I/O functions
PRMCEB	Create permanent common event flag clusters
PRMGBL	Create permanent global clusters
PRMMBX	Create permanent mailboxes
PSWAPM	Change process swap mode
SETPRV	Grant a created process any privileges
SHMEM	Create or delete data structures in shared memory
SYSGBL	Create system global sections
SYSNAM	Enter names in the system logical name table
SYSPRV	Access files and other resources as if the user has a system UIC
TMPMBX	Create temporary mailboxes
VOLPRO	Override protection on a volume
WORLD	Control the execution of any process in the system

Table 1-4
Resource Quotas

Name	Quota
ASTLM	AST (Asynchronous System Trap) limit
BIOLM	Buffered I/O limit
BYTLM	Buffered I/O byte count (buffer space) quota
CPUTIME	CPU time limit
DIOLM	Direct I/O limit
FILLM	Open file quota
PGFLQUOTA	Paging file quota
PRCLM	Subprocess quota
TQELM	Timer queue entry quota
WSDEFAULT	Default working set size
WSQUOTA	Working set size quota

CHAPTER 2

FILE SPECIFICATIONS AND LOGICAL NAMES

A file is a logically related collection of records. All the information that the operating system reads and writes on behalf of users' requests is defined in terms of files and records.

Files are identified by the hardware device that performs the actual data transfer (reading or writing). Devices are classified as:

- Mass storage devices
- Record-oriented devices

Mass storage devices provide a way to save the contents of files on a magnetic medium, called a volume. Files that are thus saved can be accessed at any time and updated, modified, or reused. Disks and tapes are mass storage devices.

Record-oriented devices read and write only single physical units of data at a time, and do not provide online storage of the data. Terminals, printers, and card readers are record-oriented devices. Printers and card readers are also called unit record devices.

This chapter discusses:

- How to specify devices and files when you enter DCL commands
- How to construct and use logical names to refer to files and devices

You can find additional specific information about how to use DCL commands to manipulate files in the command descriptions in Part II. Chapter 3, "Disk and Tape Volumes," provides more information on how to handle files on mass storage devices.

2.1 FILE SPECIFICATIONS

File specifications provide the system with all the information it needs to identify a unique file or device.

File specifications have the format:

```
node::device:[directory]filename.type;version
```

The punctuation marks and brackets are required to separate the fields of the file specification. The fields are:

Field	**Contents**
node	Network node name
device	Device name
directory	Directory name or list
filename	File name
type	File type
version	File version number

The maximum size of a file specification, including all delimiters, is 128 characters.

Directory names, file names, file types, and version numbers apply only to files on mass storage devices. For record-oriented devices, only the device name field in the file specification is required.

Additional notes and syntax requirements for each field in a file specification are discussed below. Note that you do not have to enter a complete file specification each time you specify a file; the system supplies defaults for unspecified fields. Section 2.1.5, "Defaults for File Specifications," describes defaults in more detail.

2.1.1 **Network Nodes**

If your system is part of a network, a node name may be included in a file specification to identify the computer on which the file is located.

A node name is a 1- through 6-alphanumeric character name that identifies the location on the network. Node names must contain at least one alphabetic character. If you specify a node name, you can optionally include a 3- through 42-character access control string enclosed in quotation marks (") in the format:

```
     node"access-control-string"::["]file-spec["]
```

Whether or not you specify an access control string, the node name must terminate with a double colon (::).

An access control string specifies that a particular account on the remote node should perform the file operation rather than the default network account. For VMS, the access control string consists of a user name, followed by one or more blanks or tabs and a password. For example:

```
     STAR "HIGGINS HENRY"::
```

The remainder of the file specification is passed to the remote node and is interpreted there. If the remote system requires a file specification that does not conform to the VAX/VMS syntax, you must enclose the file specification in quotation marks (").

You may also want to enclose the file specification in quotation marks to specify a task specification string that identifies a program to access on the remote node rather than a file. Thus, the three forms of network file specification, as follows:

```
node::device:[directory]filename.type;version
node::"foreign-file-spec-string"
node::"task-spec-string"
```

Optionally, a logical node name may be used in place of the node name, provided that its equivalence string represents another node specification.

For details on the syntax of file specifications and information on using DCL commands for network operations, see the DECnet-VAX User's Guide.

2.1.2 Devices

Each physical device known to the system is uniquely identified by a device name specification in the format:

```
devcu
```

where dev is a code for the device type, c is a controller designation and u is a unit number.

Table 2-1 lists the valid device types and their codes.

The controller designation and unit number identify the location of the actual device within the hardware configuration of the system. Controllers are designated with alphabetic letters A through Z. Unit numbers are decimal numbers from 0 through 65535.

The maximum length of the device name field, including the controller and the unit number, is 15 characters. When you specify a device name, terminate it with a colon (:).

Table 2-1
Device Names

Code	Device Type
CR	Card Reader
CS	Console Storage Device
DB	RP04, RP05, RP06 Disk
DD	TU58, Cassette Tape
DL	RL02 Cartridge Disk
DM	RK06, RK07 Cartridge Disk
DR	RM03, RM05 Disk
DY	RX02 Floppy Diskette
LA	LPA11-K Laboratory Peripheral Accelerator
LP	Line Printer
MB	Mailbox
MS	TS-11 Magtape
MT	TE16, TU45, TU77 Magnetic Tape
NET	Network Communications Logical Device
OP	Operator's Console
RT	Remote Terminal
TT	Interactive Terminal
XF	DR32 Interface Adapter
XJ	DUP11 Synchronous Communications Line
XM	DMC11 Synchronous Communications Line

A complete device name specification is called a physical device name. You can specify physical device names to indicate an input or output device for a command or program; or you can equate a physical device name to a logical name and use a logical name to refer to a device. Logical names are described in detail in Section 2.2.

When you refer to a file on a disk volume set, you must specify either the name of the device on which the first volume in the set is mounted or the logical name assigned to the volume set when it was mounted.

Some commands allow you to specify a generic device name. A generic device name is one in which the controller and/or the unit number are not specified. When you use a generic device name, the system locates an available device-unit whose physical name satisfies the portions of the generic device name that are specified. For example, if you issue an ALLOCATE command and specify only a device type, the ALLOCATE command locates an available unit of that type.

For all commands, except the ALLOCATE command, if you omit the controller designation, it is assumed to be A; if you omit the unit number, it becomes 0. When you omit the controller or unit number in an ALLOCATE command, the device is treated as a generic device name as just described.

2.1.3 Directories

A directory is a file that identifies (by names and locations) a set of files on a disk volume set. Directory names apply only to files on disk devices. Directory names have three possible formats:

- A 1- through 9-alphanumeric character string
- A two-part octal number in the format of a user identification code (UIC)
- A sequence of directory names (name1.name2.name3) where each name represents a directory level. Each directory name may consist of up to 9 alphanumeric characters, and you may concatenate a maximum of 8 directory names in total.

All these formats require the directory name to be enclosed in either square brackets ([and]) or angle brackets (< and >).

2.1.3.1 Directory Names

An alphanumeric directory name can be any character string that you request or that the system manager gives you. For example:

```
[MALCOLM]
```

To specify a directory name in UIC format, separate the group number from the member number with a comma. For example:

```
[122,1]
```

Directories in UIC format generally, but not necessarily, correspond to the UIC of the owner of the directory.

UIC directories can also be expressed in alphanumeric directory format. In this case, the group and member numbers are each zero-filled on the left (if necessary). For example:

```
    [122001]
```

This directory specification is equivalent to the specification [122,1] in the preceding example.

To specify a subdirectory, separate directory level identifiers with periods. For example:

```
    [MALCOLM.TESTFILES]
    [122001.TESTFILES.DATA]
```

A directory name may not mix UIC format and subdirectory format. For example, [122,1.SUB] is invalid. If you have a UIC directory, you must specify it in alphanumeric format when you are specifying its subdirectories. For example, [122001.SUB] would be valid.

2.1.3.2 **Directory Hierarchies** - You must have at least one directory, provided by the system manager, before you can create and catalog files on disks. Optionally, you can create, in your own directory, one or more directory level hierarchies.

The CREATE/DIRECTORY command can create a subdirectory. For example, the following command creates a second-level directory in the directory named MALCOLM:

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

This command places an entry for the directory file SUB.DIR in the first-level directory MALCOLM. Subsequently, you can use the subdirectory name [MALCOLM.SUB] in a file specification.

A subdirectory can contain an entry for another directory; that directory can contain an entry for another directory, and so on. The maximum number of levels, including the first-level directory, is eight. This structure constitutes a directory hierarchy. Figure 2-1 illustrates directory hierarchies.

There is no maximum number of hierarchies of directories you can create and access beginning with your own directories. However, you are confined to the amount of disk space available to you, and you must not create more than eight levels of directories in any one hierarchy.

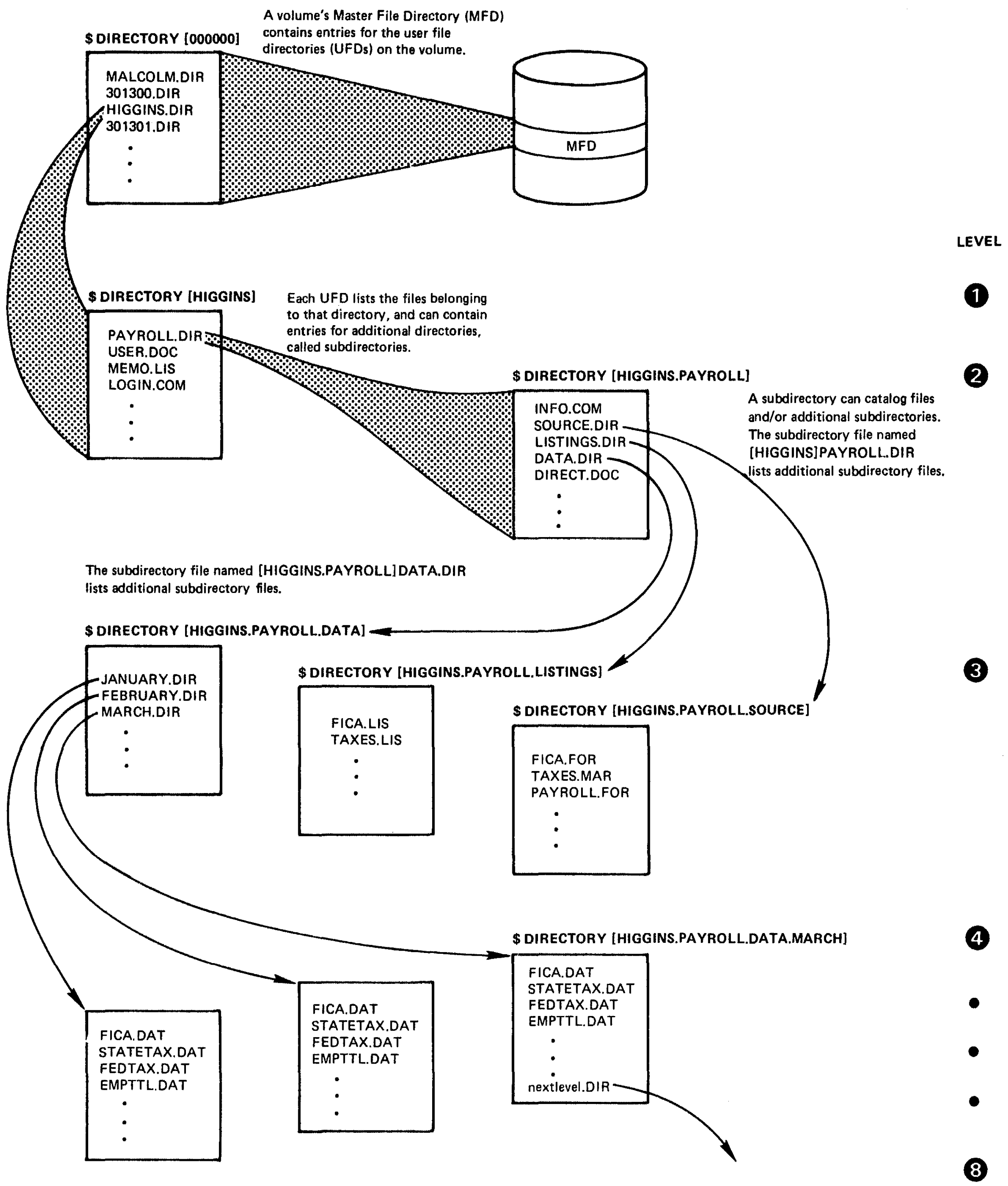

$ DIRECTORY [000000]
MALCOLM.DIR
301300.DIR
HIGGINS.DIR
301301.DIR
A volume's Master File Directory (MFD) contains entries for the user file directories (UFDs) on the volume.
MFD
LEVEL
1
$ DIRECTORY [HIGGINS]
PAYROLL.DIR
USER.DOC
MEMO.LIS
LOGIN.COM
Each UFD lists the files belonging to that directory, and can contain entries for additional directories, called subdirectories.
2
$ DIRECTORY [HIGGINS.PAYROLL]
INFO.COM
SOURCE.DIR
LISTINGS.DIR
DATA.DIR
DIRECT.DOC
A subdirectory can catalog files and/or additional subdirectories. The subdirectory file named [HIGGINS]PAYROLL.DIR lists additional subdirectory files.
The subdirectory file named [HIGGINS.PAYROLL]DATA.DIR lists additional subdirectory files.
$ DIRECTORY [HIGGINS.PAYROLL.DATA]
JANUARY.DIR
FEBRUARY.DIR
MARCH.DIR
$ DIRECTORY [HIGGINS.PAYROLL.LISTINGS]
FICA.LIS
TAXES.LIS
$ DIRECTORY [HIGGINS.PAYROLL.SOURCE]
FICA.FOR
TAXES.MAR
PAYROLL.FOR
3
$ DIRECTORY [HIGGINS.PAYROLL.DATA.MARCH]
FICA.DAT
STATETAX.DAT
FEDTAX.DAT
EMPTTL.DAT
nextlevel.DIR
4
FICA.DAT
STATETAX.DAT
FEDTAX.DAT
EMPTTL.DAT
FICA.DAT
STATETAX.DAT
FEDTAX.DAT
EMPTTL.DAT
8

Figure 2-1 A Directory Hierarchy

2.1.4 File Names, File Types, and Version Numbers

File names, file types, and version numbers uniquely identify files within directories.

A file name is a 1- through 9-character string name for a file. When you create a file, you can assign it a file name that is meaningful to you.

A file type is a 1- through 3-character string that usually identifies the file in terms of its contents.

The valid characters in file names and file types are A through Z, a through z, and 0 through 9.

File types must be preceded with a period (.).

By convention, VAX/VMS uses a set of standard file types to identify various classifications of files and to provide default file types in many commands. Table 2-2 lists default file types.

Version numbers are decimal numbers from 1 to 32767 that differentiate between versions of a file. When you update or modify a file and do not specify a version number for the output file, the system saves the original version for back-up and increments the version number by 1. Note, however, that on Files-11 Structure Level 2 disks, the file system deletes the lowest numbered versions of a file after more than approximately 60 versions of the file exist. The /VERSION_LIMIT qualifier on the CREATE/DIRECTORY command allows you to further limit this number, if desired.

Version numbers must be preceded with a semicolon (;) or a period (.).[1]

Table 2-2
Default File Types

File Type	Contents
ANL	Output file for the ANALYZE command
BAS	Input source file for the VAX-11 BASIC compiler
B2S	Input source file for the PDP-11 BASIC-PLUS-2/VAX compiler
B32 or BLI	Input source file for the VAX-11 BLISS-32 compiler
CBL	Input file containing source statements for the PDP-11 COBOL-74/VAX compiler
CMD	MCR command language and RSX-11M utility indirect command file
COB	Input file containing source statements for the VAX-11 COBOL-74 compiler
COM	Command procedure file to be executed with the @ (Execute Procedure) command, or to be submitted for batch execution with the SUBMIT command
COR	Input source file for the VAX-11 CORAL 66 compiler
DAT	Input or output data file

(continued on next page)

1. When the system displays file specifications, it generally displays a semicolon in front of the file version number.

Table 2-2 (Cont.)
Default File Types

File Type	Contents
DIF	Output listing created by the DIFFERENCES command
DIR	Directory file
DIS	Distribution list file for the MAIL command
DMP	Output listing created by the DUMP command
EDT	Initialization command input file for the EDT editor
EXE	Image file created by the linker
FOR	Input file containing source statements for the VAX-11 FORTRAN compiler
HLB	Help text library file
HLP	Input source file for help libraries
JNL	Journal file output for the VAX-11 PATCH utility
JOU	Journal file/audit trail for the EDT editor
L32	VAX-11 BLISS-32 precompiled library
LIB	Input file containing VAX-11 COBOL-74 source statements to be copied into another file during compilation
LIS	Listing file created by a language compiler or assembler; default input file type for PRINT and TYPE commands
LOG	Batch job output file
LST	Compatibility mode utility listing file
MAC	MACRO-11 source file
MAI	Mail message file
MAP	Memory allocation map created by the linker or RSX-11M Task Builder
MAR	VAX-11 MACRO source file
MLB	Macro library
MSG	Source file that specifies the text of messages
OBJ	Object file created by a language compiler or assembler
ODL	Overlay description file (RSX-11M Task Builder, only)

(continued on next page)

Table 2-2 (Cont.)
Default File Types

File Type	Contents
OLB	Object module library
OPT	Options file for input to the LINK command
PAR	SYSGEN parameter file
PAS	Input source file for the VAX-11 PASCAL compiler
R32 or REQ	VAX-11 BLISS-32 source files required for compilation
STB	Symbol table file created by the linker
SYS	System image
TLB	Text library
TMP	Temporary file
TXT	Input file for text libraries or MAIL command output
UPD	Update file of changes for a VAX-11 MACRO source program; also input to the SUMSLP editor

2.1.5 Defaults for File Specifications

When you enter a file specification, you can omit fields in the specification and let the system supply values for these fields. The values supplied by the system are called defaults.

The device and directory names, if omitted, default to your current default disk and directory name. These are initially established when you log in, based on an entry under your user name in the system authorization file.

You can determine your default disk and directory name by issuing the SHOW DEFAULT command. For example:

```
$ SHOW DEFAULT
  DBA1:[RABBIT]
```

This response indicates that the current default disk is DBA1 and the directory name is RABBIT. You can change the disk and directory defaults during a terminal session or in a batch job by using the SET DEFAULT command. For example, if the default disk and directory are as shown in the response to the SHOW DEFAULT command above, the following SET DEFAULT command would change the default directory to a subdirectory:

```
$ SET DEFAULT [RABBIT.SUB]
```

The device and directory are not the only fields for which the system supplies default values. Table 2-3 summarizes the defaults, if any, applied to each field in a file specification.

Table 2-3
File Specification Defaults

Field	Defaults
node	Local system.
device	Device (usually a disk) established at login, or by the SET DEFAULT command. If a controller designation is omitted, it defaults to A. If a unit number is omitted, it defaults to 0. (The ALLOCATE and SHOW DEVICE commands, however, treat a device name that does not contain controller and/or unit numbers as a generic device name. For more details, see the discussions of these commands in Part II.)
directory	Directory name established at login or by the SET DEFAULT command. A directory name that begins with a period (.) is appended to the current default. For example, if you specify the directory as [.SUB] when your default is [RABBIT], the effect is to reference the directory [RABBIT.SUB].
file name	No defaults are applied to the first file name in an input file specification. Most commands apply default output file names based on the file name of an input file.
file type	Various commands apply defaults for file types, based on the standard file type conventions summarized in Table 2-2.
file version	For input files, the system assumes the highest version number. For output files, if no file exists (in this directory) with the specified file name and file type, the file is created with a version number of 1. However, if one or more versions do exist, the next highest version number is used.

2.1.5.1 **Temporary Defaults** - All DCL commands that accept lists of input files apply temporary defaults when you enter a command line that contains more than one input file specification. Temporary defaulting lets you name a device, directory, file name, or file type that all the files you specify have in common. The system uses temporary file specification defaults only to interpret file specifications for a single execution of a command. Temporary defaults are applied to:

- Node name
- Device name
- Directory name
- File name and file type

If a file specification explicitly includes a device and/or directory name, these specifications become the temporary defaults for the interpretation of subsequent file specifications within the list.

File names and file types can also be defaulted, depending on the specific command.

For example, assume that the current default disk device and directory name are DBB2:[MONROE]. The following PRINT command shows how temporary defaults are applied to a list of file specifications in a parameter:

```
$ PRINT DBA1:[ADAMS]TEST1.DAT, -
$_  TEST2, -
$_  [JACKSON]SUMMARY.TST, -
$_  DBB2:FINAL
```

This example illustrates the use of the hyphen (-) as a continuation character and the automatic response of the continuation prompt ($_) that it evokes.

This PRINT command prints the files:

```
DBA1:[ADAMS]TEST1.DAT
DBA1:[ADAMS]TEST2.DAT
DBA1:[JACKSON]SUMMARY.TST
DBB2:[JACKSON]FINAL.TST
```

To override a temporary default to specify your current default directory, specify the directory as brackets with no directory name as shown below:

```
$ PRINT  [ALPHA]TEST.DAT,[]FINAL
```

In the second file specification above, the empty brackets that precede FINAL indicate the system is to use your current default directory to locate FINAL.DAT for printing.

2.1.5.2 **Null File Names and File Types** - The file name and file type fields of a file specification can be null. For example, the following are valid file specifications:

```
.TMP   ←— (file name is null)

TEMP.  ←— (file type is null)
```

When you specify a file in a DCL command, you must be careful to omit the period following a file name if the command uses a default file type. For example, the FORTRAN command uses a default file type of FOR. The following commands produce different results:

```
$ FORTRAN TEMP
$ FORTRAN TEMP.
```

In the first example, the FORTRAN compiler looks for a file named TEMP.FOR because the file type was omitted. In the second example, the compiler looks for a file named "TEMP." because a period following the file name explicitly specifies the null file type.

2.1.6 Wild Card Characters

Many DCL commands accept special "wild card" characters in their input file specifications. There are two general characters that are referred to as wild card characters: the asterisk (*) and percent sign (%). The purpose of wild card characters is to refer to a group of files by a general name, rather than by each specific name. You can combine the two wild card characters in many ways. Both wild card characters can be used in the directory specification, file name, and file type. However, only the asterisk can be specified in file version number fields.

In addition to the two general purpose wild card characters, there are two special wild card characters that can be used only in alphanumeric directory specification fields. These are the sequence of three dots known as an ellipsis (...) and the minus sign (-).

When a command allows wild card characters in all parts of the file specification, it is said to allow full wild cards, or full wild carding.

Particular uses of wild card characters in DCL commands vary with the individual commands. The command descriptions in Part II of this manual indicate (in each place where a file specification is described) whether wild card characters are allowed.

The next three subsections describe the four wild card characters and how they apply to file specifications for input files only. The fourth subsection describes a special use of the asterisk (*) in output file specifications.

NOTE

Special rules and limitations exist for wild card characters in file specifications for network operations. See the DECnet-VAX User's Guide.

2.1.6.1 **The Match-All Wild Card Character** - The asterisk (*) wild card character indicates that you want to process all file specifications that match any possible value in this field or portion of the field.

The number of characters allowed for this field ranges from zero to the maximum size field. That is, if the field containing the asterisk permits three characters, as the file type does, matches occur on fields that are zero, one, two, or three characters long.

Consider some examples of the asterisk used in place of an entire field. The file specification that will select all versions of all files in a directory is shown in this COPY command:

```
     COPY [MALCOLM]*.*;* DMA1:[BACKUP.JULY]*.*;*
```

The COPY command generally copies the contents of a specified input file into a new output file. With this wild card character capability, you can copy large numbers of files without naming them individually. In this case, all the files in the directory named MALCOLM are copied to a directory named BACKUP.JULY on an RK07 device. The file specification with wild card characters is useful in many operations, and a copy operation is but one of them.

A file specification such as the following limits the files selected to a more specific group:

```
*.DAT;*
```

Note that the file specification above selects only those files with file types of DAT out of all the files in the current default disk and directory.

Consider another example where wild card characters appear in the directory specification:

```
[*.*.*]AVERAGE.*;*
```

With this file specification you get all versions of all files named AVERAGE, with any file type, that exist on any directory on the current default disk down to two sublevels of each of those directories.

It is also possible to use the asterisk wild card character in directory specifications given in the UIC format (Section 2.1.3.1). For example, [*,6] locates all directories with any group number and a member number of 6. Since the search is limited to directories in UIC format, a directory specification of [*,*] does not include any of the alphanumerically named directories. However, [*] locates all alphanumerically named directories and all directories in the UIC format, too.

The asterisk wild card character can also be used to match all characters in any portion of a file specification field (except the version number), whether at the beginning, middle, or end. In fact, you can have several asterisks in one field if you need to. Assume you had named all your subroutines for a tax program beginning with the letters STAX, followed by 0 to 5 additional alphanumeric characters. You could select this group of files with a file name specification of:

```
STAX*
```

The more complex file specification *INS*9*.D*;1 (which uses multiple asterisks) illustrates how to select such diverse files as: AINST95.DAT;1, INS9.D;1, and COBINS90A.DIR;1.

2.1.6.2 **The Match-Any-Character Wild Card Character** - The percent sign (%) allows you to select files with any single character in the position that the percent sign occupies in the file specification. For example:

```
[MALCOLM]CHAP%.DOC;*
```

This file specification selects all versions of all files with a file type of DOC in the directory named MALCOLM that have file names beginning with CHAP followed by a single character. Some possibilities that might be selected include CHAPA.DOC, CHAP3.DOC, and so forth. Note, however, that CHAP.DOC is not selected since it contains nothing in the percent sign position. Likewise, CHAPIX.DOC is ignored because it has too many characters after CHAP in its file name; only one position is reserved by the percent sign here.

You can specify the percent sign in any part of a file specification, as many times as necessary, and in combination with other wild card characters, too. Thus, if you want to select all versions of files having file types beginning with J and file names starting with INS followed by any three characters before an A, in directories whose names begin with MA, you might issue the following file specification:

```
[MA*]INS%%%A*.J*;*
```

Some of the files that would be selected by the file specification above include:

```
[MAINE]INS123A.JNL;1
[MASSACHUS]INS854A89.JTK;43
[MALCOLM]INS743A9.J;13
[MANDELL]INS912A75.JC;24
```

2.1.6.3 **The Directory Searching Wild Card Characters** - The ellipsis (...) and minus sign (-) serve special purposes in alphanumeric directory specifications. These wild card characters are aids to searching, or traversing, directory hierarchies. Directory hierarchies are described in Section 2.1.3.2. Both the ellipsis and minus sign, in addition to the period, allow you to refer to directories in a relative positional sense, rather than by an absolute name for the first directory or group of directories.

Searching Down

The ellipsis indicates that you want to select files from all directory levels from a specified level downward to lower levels of the hierarchy. This means that a file specification such as

```
[JONES...ANALYSIS]PQUEST.*;*
```

would search for all files named PQUEST in any of the subdirectories named ANALYSIS under the directory JONES.

You must understand the concept of the default directory hierarchy to learn additional features of directory searching. The default directory is defined as the result of the last SET DEFAULT command.

If you want to start a directory search from your default position, you can begin the directory specification with a period, an ellipsis, or a minus sign.

For example, the specification:

```
[...BCOMP]
```

matches all directories named BCOMP below the default directory position.

If you want to search all the way to the bottom of the directory hierarchy, you can use a directory specification that ends with an ellipsis. For example:

```
[...INVENTORY...]ZSTOCK.DAT
```

This specification locates all files named ZSTOCK.DAT in all directories from the directory named INVENTORY below the default level down to the bottom of the INVENTORY hierarchy. Note that in this case INVENTORY must occur on a path down from the current position, and it must be at least one level below the current position. This example also illustrates the use of multiple ellipses in one directory specification.

```
[...INVENTORY...*]ZSTOCK.DAT
```

In this case, however, the asterisk requires that ZSTOCK.DAT occurs in at least one level of subdirectory below INVENTORY. In the previous example, a match would have occurred if ZSTOCK.DAT had been found in the INVENTORY directory itself. You will find that you can combine all four wild card characters in directory specifications in numerous ways.

You can specify a search of all directories and subdirectories on the volume with

```
     [*...]
```

This specification searches down as many as eight levels of directory names, if they exist.

If you want to traverse all subdirectories below your default directory, [...] will suffice.

Searching Up

Sometimes you need to search up the hierarchy rather than down. A single minus sign will send the search back up one level from the default directory level. Thus, if the default position in the directory hierarchy is

```
     [JONES.TEST.BACKUP.SRC]
```

you could print the file [JONES.TEST.BACKUP.COM]HUMIDITY.LIS with the command

```
     PRINT [-.COM]HUMIDITY.LIS
```

Also note that more than one minus sign can be used. For example, if instead you specify

```
     [--.COMPUTE]
```

you traverse back up the same default hierarchy to directory TEST and then down to its subdirectory COMPUTE or, in other words, to [JONES.TEST.COMPUTE].

Be careful not to issue so many minus signs that you point above the top directory level; remember that VMS permits up to eight levels of directory names.

2.1.6.4 **Temporary Defaults in Output Files** - Only one of the wild card characters, the asterisk, is allowed in output file specifications. When used in an output specification, the asterisk indicates a temporary default is desired. That is, you want the selection of the output files to follow the corresponding field in the input specification. Some commands apply their own defaults to the output file type when it is absent. Section 5.3.3 describes all the rules that govern the resolution of defaults in output file specifications.

The following examples illustrate how DCL interprets asterisks in the output file specifications of COPY commands.

Example	**Explanation**
COPY TEST.DAT *.OLD	Copies the highest version of the file TEST.DAT from the current default disk and directory into a file named TEST.OLD
COPY [CHEVY]*.FOR *	Copies the highest version of each file with a file type of FOR in the directory CHEVY on the current default disk to new files in the current default directory

2.2 LOGICAL NAMES

Logical names allow you to keep programs and command procedures independent of physical file specifications. They also provide a convenient shorthand way to specify files that you refer to frequently.

The ASSIGN command equates a physical file specification to a logical name, that is, to a character string name that you supply. For example:

```
$ ASSIGN DBA2:[SIMMONS]MARIGOLD.DAT  TEST
```

This ASSIGN command equates the logical name TEST to the file specification DBA2:[SIMMONS]MARIGOLD.DAT. This file specification is called the equivalence name for the logical name. Subsequently, you can refer to this file by its logical name when you issue a DCL command. For example:

```
$ TYPE TEST
```

When the system processes this TYPE command, it replaces the logical name TEST with its equivalence name and displays the contents of the file MARIGOLD.DAT on the terminal.

You can also assign logical names to devices when you issue ALLOCATE, DEFINE, or MOUNT commands.

2.2.1 Logical Name Tables

The system maintains logical name and equivalence name pairs in three logical name tables:

- Process logical name table -- contains logical name entries that are local to a particular process. By default, the ASSIGN command places a logical name in the process logical name table.
- Group logical name table -- contains logical name entries that are qualified by a group number. These entries can be accessed only by processes that execute with the same group number in their user identification codes as the process that assigned the logical name. You must use the /GROUP qualifier to make an entry in the group logical name table.

- System logical name table -- contains entries that can be accessed by any process in the system. You must use the /SYSTEM qualifier to make an entry in the system logical name table.

The user privileges GRPNAM and SYSNAM are required to place entries in the group or system logical name tables, respectively.

2.2.2 How to Specify Logical Names

Logical names and their equivalence name strings can each have a maximum of 63 characters, and can be used to form all or part of a file specification. If only part of a file specification is a logical name, it must be the left-most component of the file specification. You can then specify the logical name in place of the device name in subsequent file specifications, terminated by a colon (:).

For example, a logical name can be assigned to a device name, as follows:

```
$ ASSIGN DMA1: BACKUP
```

After this ASSIGN command, you can use the logical name BACKUP in place of the device name field when referring to the device. For example, the following COPY command transfers files from the current default disk and directory to a directory with the same name on the volume on the device DMA1 by using the logical name BACKUP.

```
$ COPY *.* BACKUP:
```

A logical name can also contain both a device name and a directory name. For example:

```
$ ASSIGN DMA1:[MAGGIE] SCRATCH
```

This ASSIGN command assigns the logical name SCRATCH to the directory named MAGGIE on the device DMA1. You can now use the logical name SCRATCH in a file specification, as in the example below.

```
$ PRINT SCRATCH:PAYROLL.DAT
```

The PRINT command prints the file PAYROLL.DAT from the directory [MAGGIE] on DMA1.

When you specify an equivalence name for the ASSIGN command, you must specify it using the proper punctuation marks (colons, brackets, periods). If you specify only a device name, terminate the equivalence name parameter with a colon (:); if you specify a device and directory name, or a full file specification, do not terminate the equivalence name with a colon.

You can optionally terminate the logical name parameter with a colon; however, the ASSIGN command removes the colon before placing the name in the logical name table.[1]

1. The DEFINE command, which also creates logical name table entries, does not remove colons, if specified, from logical names. It is recommended that you do not use the DEFINE command for device name assignments.

2.2.2.1 **Displaying Logical Name Table Entries** - The SHOW LOGICAL command displays current entries in the logical name tables. For example, to display the logical name SCRATCH, you would enter the command:

```
$ SHOW LOGICAL SCRATCH
  SCRATCH  =  DMA1:[MAGGIE]      (process)
```

The SHOW LOGICAL command displays the logical name, its equivalence name, and identifies the logical name table in which it found the logical name. In this example, the logical name SCRATCH occurs in the process logical name table with the equivalence name of DMA1:[MAGGIE].

You can also request the system to display all the entries in a specified logical name table. For example:

```
$ SHOW LOGICAL/GROUP
```

This SHOW LOGICAL command results in a display of all current entries in the group logical name table.

2.2.3 Logical Name Translation

When the system reads a device name or a file specification, it examines the file specification to see if the left-most component is a logical name. If it is, the system substitutes the equivalence name in the file specification. This is called logical name translation.

For example:

```
$ TYPE ALPHA
$ TYPE DISK:ALPHA
```

When the system reads the file specification ALPHA in the first example, it checks to see if ALPHA is a logical name because ALPHA is the left-most (and in this example, the only) component of the file specification. In the second example, the system checks to see if DISK is a logical name because it is the left-most component. It does not check ALPHA.

When the system translates logical names, it searches the process, group, and system tables, in that order, and uses the first match it finds.

If you are ever in doubt about the current equivalence name assigned to a logical name, use the SHOW LOGICAL command.

2.2.3.1 **Recursive Translation** - When the system translates logical names in file specifications, the logical name translation can be recursive. This means that after the system translates a logical name in a file specification, it repeats the process of translating the file specification. For example, consider logical name table entries made with ASSIGN commands as follows:

```
$ ASSIGN DBA1:  DISK
$ ASSIGN DISK:WEATHER.SUM REPORT
```

The first ASSIGN command equates the logical name DISK to the device DBA1. The second ASSIGN command equates the logical name REPORT to the file specification DISK:WEATHER.SUM. In subsequent commands, or in programs you execute, you can refer to the logical name REPORT. For example:

```
     $ TYPE REPORT
```

When the system translates the logical name REPORT, it finds the equivalence name DISK:WEATHER.SUM. It then checks to see if the portion to the left of the colon in this file specification is a logical name; if it is (as DISK is in this example), it translates that logical name also. When the logical name translation is complete, the translated file specification is:

```
     DBA1:WEATHER.SUM
```

Note that when you assign one logical name to another logical name, you must terminate the equivalence name with a colon (:) if you are going to use the logical name in a file specification in place of a device name. For example:

```
     $ ASSIGN DBA1:  TEST
     $ ASSIGN TEST:  GO
     $ TYPE GO:NEW.DAT
```

The TYPE command types the file NEW.DAT from the disk DBA1. If you omit the colons from either of these ASSIGN commands, the system will not be able to form a proper file specification.

The system limits logical name translation to ten levels. If you define more than ten levels or create a circular definition, an error occurs when the logical name is used.

2.2.3.2 **Applying Defaults** - When the system completes the translation of a logical name, it uses defaults to fill in any still unspecified fields in the file specification. In the above examples, the system completes the file specifications by supplying the current default directory and a version number according to whether it is an input or an output file.

Many system commands create output files automatically and provide default file types for the output files. When you use a logical name to specify the input file for a command, the command uses the logical name to assign a file specification to the output file as well. Thus if the equivalence name contains a file name and file type, the output file is given the same file name and file type as the input file but a higher version number.

For example, the LINK command creates, by default, an executable image file that has the same file name as the input file and a default file type of EXE. If you make a logical name assignment as shown below and invoke the LINK command, the results may not be what you expect:

```
     $ ASSIGN RANDOM.OBJ TESTIT
     $ LINK TESTIT
```

In this example, the translation of the logical name TESTIT provides the file RANDOM.OBJ as input to the linker. When the linker creates the output file, it also uses the same logical name for the output file. Because the equivalence name has a file type, the default file type of EXE is ignored. Instead, the executable image is named RANDOM.OBJ and it has a version number one higher than the version number of the input file.

2.2.3.3 **Logical Names in Input File Lists** - When you issue a command that accepts multiple input files and you use logical names in the specifications of one or more files in the list, the equivalence name of each logical name provides a temporary default. For example:

```
$ SET DEFAULT  DBA2:[CASEY]
$ ASSIGN  DBA1:[MALCOLM] MAL
$ ASSIGN  [HIGGINS] HIG
$ PRINT  ALPHA,-
$_ MAL:BETA,-
$_ HIG:GAMMA
```

The PRINT command looks for the files:

```
DBA2:[CASEY]ALPHA.LIS
DBA1:[MALCOLM]BETA.LIS
DBA1:[HIGGINS]GAMMA.LIS
```

The device name in the equivalence string for the logical name MAL defines DBA1 as the temporary default device for this PRINT command. Temporary defaults are described in Section 2.1.5.1. For complete details on file specification defaults, see the VAX-11 Record Management Services Reference Manual.

2.2.3.4 **Bypassing Logical Name Translation** - Most DCL commands check file specifications you enter as command parameters or as values for qualifiers to see if the file specification contains a logical name. When you enter a device name in a command, you can request the system not to translate the name by preceding the device name or file specification with an underscore character (_). For example:

```
$ ALLOCATE _DMA2:
```

When you specify an ALLOCATE command as shown, the system does not attempt translation of DMA2.

2.2.4 Default Logical Names

The operating system and many of its facilities use logical names to establish default devices for input and output operations. By convention, logical names defined for VAX/VMS functions have the format:

```
xxx$name
```

where xxx is a three-character prefix identifying the system component that uses the logical name.

2.2.4.1 **Default Process Logical Names** - When you log in to the system, the system creates logical name table entries for your process. These logical names, which all have a prefix of SYS, are listed in Table 2-4.

Table 2-4
Default Process Logical Names

Logical Name	Equivalence Name
SYS$COMMAND	Original (that is, first level) SYS$INPUT stream.
SYS$DISK	Default device established at login, or changed by the SET DEFAULT command. SYS$INPUT Default input stream for the process. For an interactive user, SYS$INPUT is equated to the terminal. In a command procedure or batch job, SYS$INPUT is equated to the current input stream.
SYS$ERROR	Default device to which the system writes messages. For an interactive user, SYS$ERROR is equated to the terminal. In a batch job, SYS$ERROR is equated to the batch job log file.
SYS$LOGIN	Device and directory established at login time as the residence of LOGIN.COM for each process. This "home" directory is specified in the authorization record.
SYS$NET	The source process that invoked the target process in DECnet task-to-task communication. When opened by the target process, SYS$NET represents the logical link over which the target process can exchange data with its partner. SYS$NET is only defined during the task-to-task communication. For additional information, see the DECnet-VAX User's Guide.
SYS$OUTPUT	Default output stream for the process. For an interactive user, SYS$OUTPUT is equated to the terminal. In a batch job, SYS$OUTPUT is equated to the batch job log file.

The equivalence names for SYS$INPUT, SYS$OUTPUT, SYS$ERROR and SYS$COMMAND define files that remain open for the life of the process. These files, called process permanent files, can be read or written from programs.

2.2.4.2 **Using Default Logical Names** - The default logical name assignments are provided for your convenience, and do not normally need to be reassigned. The system always uses SYS$INPUT, SYS$OUTPUT, and SYS$ERROR to read and write commands, data, and messages.

You can take advantage of the default assignments when you specify files in commands. For example, to place input data in the command stream for a command or program, you can specify an input file as SYS$INPUT. The following example shows a FORTRAN command that could be executed in a batch job:

```
    $ FORTRAN SYS$INPUT:WEATHER
```

When this command executes, the compiler reads the input file from the command stream; if this job is submitted through the system card reader, the cards containing the source program must follow in the card deck. The file name WEATHER in the file specification provides the compiler with a default file name for the output files: it creates WEATHER.OBJ and WEATHER.LIS.

To request that the listing file be written directly into the batch job output log file, you could specify:

```
     $ FORTRAN/LIST=SYS$OUTPUT   SYS$INPUT:WEATHER
```

The compiler creates the file WEATHER.OBJ, but prints the listing in the batch job output log.

2.2.4.3 **Default System Logical Names** - The system logical name table contains entries for system-wide logical names. Among these logical names are the default system logical names shown in Table 2-5.

Table 2-5
Default System Logical Names

Logical Name	Equivalence Name
SYS$HELP	Device and directory name of system help files
SYS$LIBRARY	Device and directory name of system libraries
SYS$MESSAGE	Device and directory name of system message files
SYS$NODE	Name of the current network node for the local system if DECnet is active on the system
SYS$SYSDISK	VMS system disk where SYS$SYSTEM resides
SYS$SHARE	Device and directory name of system shareable images
SYS$SYSTEM	Device and directory of operating system programs and procedures

The system manager at your installation can place names in the system logical name table that correspond to default devices on your particular system.

2.2.5 Logical Names for Program Input/Output

Logical names are also used to establish the correspondence between a file name or logical unit number in a program and a physical device or file specification. Each programming language provides conventions for identifying files within programs; the operating system provides the logical name mechanism for equating these files with physical device or file specifications.

For example, a FORTRAN program refers to a file using a logical unit number. FORTRAN run-time procedures associate the logical unit numbers with logical names: logical unit 1 is associated with the logical name FOR001, logical unit 2 is associated with the logical name FOR002, and so on. Before running a program that reads from or writes to logical unit 1, you can make an assignment of that logical unit to a device with the ASSIGN command, as shown in the following example:

```
     $ ASSIGN  PAYROLL.DAT  FOR001
     $ RUN FICA
```

Before you execute programs to read and write files, you must take steps to ensure that the data volumes or devices that your program requires are online and available. The next chapter discusses how to prepare and use disk and tape volumes.

CHAPTER 3

DISK AND TAPE VOLUMES

You can use your default disk directory to catalog the files that you use on a regular basis. The physical disk volume that contains your default directory contains the files belonging to other users as well; such volumes are known as system volumes. Under normal operating circumstances, you do not need to perform any special action to access your own files.

Occasionally, you may also need to access files belonging to other users, or to transfer files to or from volumes other than the system volume containing your default directory. In these cases, special action may be required.

This chapter describes:

- How the operating system protects and restricts access to files and devices
- How to initialize a volume
- How to mount volumes on devices
- How to transfer files to and from disk and tape volumes
- How to access files from batch jobs

This chapter does not describe the physical operation of disk and tape devices. Readers who are unfamiliar with the devices are referred to the appropriate hardware manuals.

3.1 PROTECTION

The operating system protects data on disk and tape volumes to ensure against accidental or unauthorized access. Protection is provided at two levels:

- At the volume and file level to ensure that data on a volume is protected
- At the device level to ensure that no other users can access a device in use by one user

3.1.1 **Volume and File Protection**

Disk and tape volumes and individual files on disk volumes, including directories, are protected by means of a protection code. The protection code indicates who is allowed access for what purposes.

Four categories of user are defined according to UIC. These are:

- SYSTEM -- all users who have low group numbers, usually from 0 through 10 (octal). However, the exact range of group numbers is determined by the system manager when the system is generated and may range as high as 377. These group numbers are generally for system managers, system programmers, and operators.
- OWNER -- the user with the same UIC as the person who created and therefore owns the volume or file
- GROUP -- all users who have the same group number in their UICs as the owner of the file, including the owner
- WORLD -- all users

Figure 3-1 illustrates the relationships of these categories to each other.

Each of these categories of user can be allowed or denied any of the following types of access:

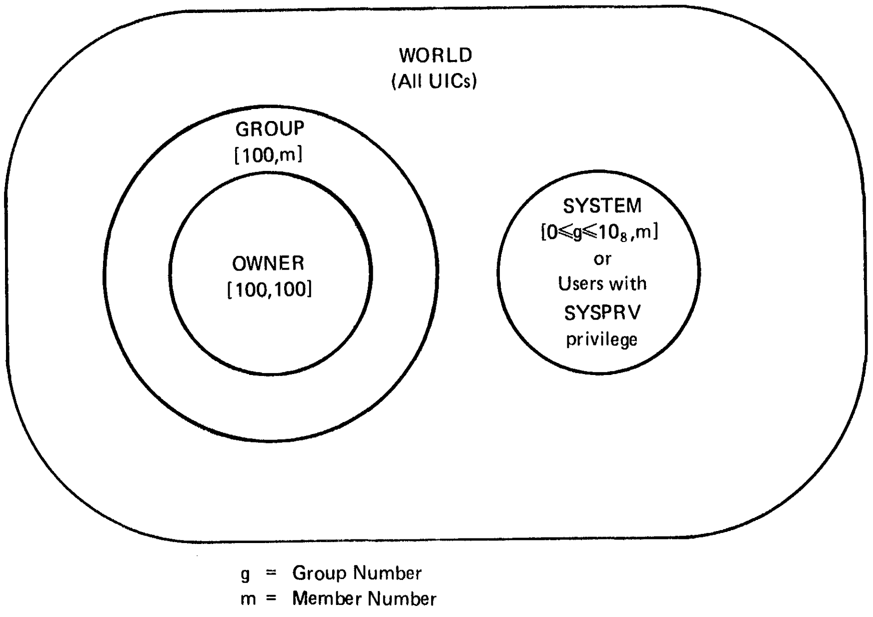
- READ -- the right to examine, print, or copy a file or files on a volume
- WRITE -- the right to modify the file or to write files on a volume
- EXECUTE -- the right to execute files that contain executable program images (when applying protection to an entire volume, this field is interpreted as the right to create files on the volume)
- DELETE -- the right to delete the file or files on the volume

The system provides a default protection code for files you create.

When you create a file or prepare a disk or tape volume for private use, you can define the protection you want to be applied. Some examples of specifying protection codes for individual files are shown in the following section; examples of volume protection are shown in Section 3.4, "Using Disk and Tape Volumes." For details on how to specify protection codes, see Section 5.10.

Each directory has a protection associated with it. The directory protection can override the protection of individual files in the directory. For example, if a directory denies WORLD access, world users cannot access even those files in the directory that permit WORLD access.

You can bypass all UIC-based protection checks if you have the BYPASS user privilege.



NOTE: THE SYSTEM MANAGER CAN EXTEND THE SYSTEM GROUP NUMBER LIMIT TO 377_8

Figure 3-1 Illustrating User Categories with a UIC of [100,100]

3.1.1.1 **Disk File Protection** - Each file on a disk has its own protection code; you can specify a protection code when you create a file. For example, you can use the /PROTECTION qualifier to define the protection for a file you create with the COPY command, as shown below:

```
$ COPY DBA1:[PAYDATA]PAYROLL.DAT  PAYSORT.DAT -
$_/PROTECTION=(SYSTEM:RW,OWNER:RWED,GROUP:RW,WORLD)
```

This COPY command copies a file from the device DBA1: to your default disk directory. The protection code defines the protection for the newly created file PAYSORT.DAT as follows: users with system UICs can read and write the file; you (the owner) have all types of access; other users in your group may read and write the file; and all other users (the world) are permitted no access.

You can also change the protection for an existing file with the SET PROTECTION command. For example:

```
$ SET PROTECTION=(SYSTEM:RWE,OWNER:RWED,GROUP:RE,WORLD) -
$_PAYSORT.EXE
```

If you do not define a protection code for a file when you create the file, the system applies a default protection. You can determine the current protection by issuing the SHOW PROTECTION command:

```
$ SHOW PROTECTION
  SYSTEM=RWED, OWNER=RWED, GROUP=RWED, WORLD=RE
```

This response is, in fact, the system default protection. It indicates that the system, owner, and group have all types of access, and that all other users (the world) are permitted read and execute access only.

To determine the current protection associated with a specific file or files, use the /PROTECTION qualifier on the DIRECTORY command. For example:

```
$ DIRECTORY/PROTECTION  PERSONNEL.REC

Directory DBA1:[CRAMER]

PERSONNEL.REC;5        (RWED,RWED,RW,R)

Total of 1 file, 8 blocks.
```

You can change the default protection applied to files that you create during a terminal session with the SET PROTECTION /DEFAULT command. The SET PROTECTION/DEFAULT command indicates that the protection code you specify is to be applied to all files that you subsequently create during the terminal session or batch job.

3.1.1.2 **Tape File Protection** - The protection applied to a tape volume applies equally to all files on the volume. The system only applies read and write access restrictions with respect to tapes; execute and delete access are meaningless. Moreover, the system and the owner are always given both read and write access, regardless of what you specify in a protection code. If you give write access to the group or world, read access is also allowed. Protection on a given tape cannot be changed.

3.1.2 Device Allocation

In most installations, many users must share a limited number of private disk drives and tape drives. The operating system controls access to these devices so that while you are using a device, no other user can access it. This control mechanism is called device allocation.

The ALLOCATE command allocates a device and gives you exclusive use of it. For example, to use the tape drive labeled MTB1, you would issue the following ALLOCATE command:

```
$ ALLOCATE MTB1:
  _MTB1: ALLOCATED
```

The response from the ALLOCATE command indicates successful allocation of the device you requested. You can also use the SHOW DEVICES command to find out what devices are available. For example:

```
$ SHOW DEVICES  DM:
```

This command requests a display of all RK06 and RK07 devices attached to the system. The response from the command shows the devices that are currently allocated to other users, and devices that are online and available.

If you want to allocate any device of a particular type, use a generic device name in the ALLOCATE command. A generic device name is one that does not specify a controller and/or a unit number. For example, if you want to use a magnetic tape device and do not know which drives are available, you would issue the ALLOCATE command as follows:

```
$ ALLOCATE MT:
  _MTA2: ALLOCATED
```

The ALLOCATE command locates an available magnetic tape device; the response indicates that the device MTA2 is allocated.

Note that tape devices can never be shared; thus, when you use a tape, the system automatically allocates it for you when you mount it. You can, however, allocate a tape device when you want to mount more than one volume on the same device and retain control of the device between mount operations.

3.2 VOLUME INITIALIZATION

Before a volume can be used to contain files, it must be initialized. The INITIALIZE command:

- Invalidates all existing data on the volume, if any, and creates a new file structure
- Writes an internal label on the volume to identify it
- Defines the owner UIC and protection for the volume

If a disk or tape volume has never been used before, no special steps are needed to initialize it. However, if the volume has previously contained data, the protection code on it may prevent you from accessing the volume and initializing it. Or, in the case of a tape, files on the volume may not have reached their expiration dates.

If either of these conditions exists and you do not have the VOLPRO user privilege to override volume protection, the previous owner of the volume or another user (the system manager or operator, for example) who does have read/write access must initialize it for you. When you give the volume to another user for initialization, you should specify:

- The data format you require
- The label you want to have written on the volume
- The protection code and owner UIC you want assigned to it

When you obtain a tape or disk volume, you should also remember to place identification on the outside of the volume so that it can be easily identified.

3.3 MOUNTING VOLUMES ON DEVICES

Mounting is the mechanism that provides a link between a volume, a device, and your process so you can perform input/output operations. Mounting and using a physical volume (a disk pack or tape reel) involves two distinct operations:

- Place the volume on the device and ready the device by pressing the load button or performing equivalent startup procedures.
- Issue the MOUNT command to gain access to the data on the device. The MOUNT command verifies the label on the volume against the label you specify.

The order in which these operations are performed varies according to the way in which the device is being used. If you have already allocated a device using an explicit ALLOCATE command, you can physically load the device before issuing the MOUNT command. For example:

```
$ MOUNT DMA2:  TEST_FILES INFILE
%MOUNT-I-MOUNTED, TEST_FILES mounted on _DMA2:
```

In this example, the MOUNT command specifies the name of an allocated device (DMA2), the volume label on the volume (TEST_FILES), and a logical name for the device (INFILE). The logical name parameter is optional. Note that you can assign logical names to disk and tape devices when you mount them, and then use the logical name rather than the device name in subsequent commands. If you do not specify a logical name, the MOUNT command assigns the default logical name DISK$TEST_FILES to the device DMA2.

3.3.1 Requesting Operator Assistance

At some installations, operators perform the physical mounting and dismounting of both system and private disk and tape volumes. When this is the case, you can:

- Allocate a device of the required type.
- Send a message to the operator specifying the name of the device you allocated and the volume you want mounted. Give the physical identification on the outside of the volume and tell the operator where the volume is.
- Wait until the operator responds with a message indicating that the volume is mounted.

The REQUEST command sends a message to an operator. If your installation has more than one operator, they may be designated for specific functions -- one operator to handle requests for disks, another for tapes, and so on. Options for the REQUEST command qualifier /TO -- for example, DISKS and TAPES -- route your request to the proper operator.

Assume, for this example, that there is an operator designated to mount and dismount disks. To request this operator to mount the volume TEST_FILES on the device DMA2, you could issue the command:

```
$ REQUEST/TO=DISKS/REPLY -
$_"Please mount TEST_FILES (shelf slot 6B) on DMA2"
```

The /REPLY qualifier indicates that you want to wait until the operator completes the request.

You receive the message:

```
%OPCOM-S-OPRNOTIF, operator notified, waiting...13:14:26
```

The operator locates the physical volume, places it on the device, and types a message indicating that the request is satisfied. Then, you receive the messages:

```
%OPCOM-S-RQSTCMPLTE, request complete
%OPCOM-S-OPREPLY, "Go..."
```

The text of the second message is optional text typed by the operator. Now, you can issue a MOUNT command and begin using the volume.

3.3.2 Dismounting Volumes

When you no longer need access to the files on a volume, you can release the volume with the DISMOUNT command. For example:

```
$ DISMOUNT DMA2:
```

The DISMOUNT command ensures that all files on the volume are closed before the dismounting is complete.

By default, the DISMOUNT command also unloads the volume on the device and makes the device not ready.

If you allocated the device before using it and no longer need the device, deallocate it so that other users can obtain access to it. The DEALLOCATE command deallocates the device. For example:

```
$ DEALLOCATE DMA2:
```

If you had operator assistance in mounting the volume, remember to request the operator to physically dismount the volume and return it to its place.

Either logging out or terminating a batch job will automatically dismount and deallocate all private volumes.

3.4 USING DISK AND TAPE VOLUMES

When a volume has been mounted, you can execute programs that perform input/output operations to the volume, or you can use DCL commands to read and write files. Note the following restrictions on the use of DCL commands for files on disk and tape volumes:

- The PRINT and SUBMIT commands cannot access files on allocated devices. You can print or submit files on private disk volumes if you mount the volume as a shareable volume. To print or submit a file on a tape volume, you must copy the file to a shared disk volume.
- The following commands require file-structured devices:

```
DELETE          PURGE
DIFFERENCES     RENAME
LIBRARY         RUN
LINK            SET PROTECTION
                UNLOCK
```

- You can execute a command procedure that exists on a magnetic tape volume, as long as the procedure does not invoke other procedures and does not issue any GOTO commands referring to labels that precede the command.

Note, also, that you cannot use DCL commands to read or write files that are not in the standard formats supported by VAX/VMS (these formats are described in greater detail, below). To execute programs to read and write files not in the standard format, you must mount the volume with the /FOREIGN qualifier.

The following sections provide complete examples of the steps to prepare and use disk and tape volumes for file storage and to access existing files. The examples show how to prepare and use RK06/RK07 disk packs and magnetic tapes; however, the procedures outlined are applicable to other devices as well.

3.4.1 **Using Disks**

Disks are random-access devices, and files must be cataloged in directories. Therefore, after you initialize a disk, you must create a directory before you can write any files on the disk volume.

The following example shows how to allocate an RK06/RK07 device, initialize and mount a volume on it, create a directory, and write files on the volume.

```
$ SHOW DEVICES DM:
  List of Devices             on        5-FEB-1980 14:56:40.18
  Device   Device     Device      Err.    Volume        Free  Trans Mount
  Name     Status  Characteristics Count   Label        Blocks Count Count
  DMA0:    on line mnt              .        .            .     .     .
  DMA1:    on line mnt all          .        .            .     .     .
  DMA3:    on line                  .        .            .     .     .

$ ALLOCATE _DMA3:
  _DMA3:  ALLOCATED
```

This example illustrates the SHOW DEVICES command requesting a display of the current status of RK06/RK07 devices. The response from the command indicates that the device named DMA3 is available. The ALLOCATE command allocates DMA3 for your exclusive use.

```
    $ INITIALIZE DMA3:  PUBS_BACKUP -
    $_/PROTECTION=(SYSTEM:RWED,OWNER:RWED,GROUP:RWED,WORLD:R)
    $ MOUNT DMA3:  PUBS_BACKUP
    $ CREATE/DIRECTORY DMA3:[PUBS]
    $ ASSIGN DMA3:[PUBS]  P:
    $ COPY *.*  P:
    $ COPY [PRIMER]*.*  P:
    $ COPY [COMMANDS]*.*  P:
```

The INITIALIZE command initializes the volume and writes the label PUBS_BACKUP on it. The protection code allows group members and users with system UICs all access and restricts access by all other users to reading. The MOUNT command mounts the volume. The CREATE/DIRECTORY command creates a directory file named PUBS on the device DMA3. The COPY commands copy the highest versions of all files in the current default directory and in the directories PRIMER and COMMANDS to the directory just created.

Note the use of the ASSIGN command to assign a logical name, P, to the device and directory name on the RK06/RK07 volume.

3.4.1.1 **Copying Files from Files-11 Structure Level 1 Disks** - The default format for files on disks is called Files-11 Structure Level 2. You can also initialize disks in the Files-11 Structure Level 1 format. Structure Level 1 is the format used by other DIGITAL operating systems, including RSX-11M, RSX-11M-PLUS, RSX-11D, and IAS. Both Files-11 formats are described in detail in the Introduction to VAX-11 Record Management Services.

To initialize a Structure Level 1 disk, use the /STRUCTURE=1 qualifier on the INITIALIZE command. When you use the MOUNT command to mount the volume, the MOUNT command will internally identify the volume as a Structure Level 1 volume. Subsequently, all commands you use to access files (COPY, DELETE, and so on) will default the file format to Structure Level 1 automatically. Note that directories on Structure Level 1 disks must have names in UIC format to be readable by RSX-11 or IAS.

3.4.1.2 **Sharing Volumes** - System disk volumes containing users' default directories and other volumes containing files belonging to more than one user are designated, at the time that they are mounted, as shareable. The devices on which these volumes are mounted are not allocated, thus allowing access by other system users. These volumes are physically loaded and mounted by system operators and managers with the /SYSTEM qualifier of the MOUNT command; you need not issue an explicit MOUNT command to access files on these volumes (you must, however, have access privileges as defined in the volume's protection code).

Other volumes can be designated as shareable among many users; use the /SHARE qualifier on the MOUNT command when you want to mount a device for other users to access. For example:

```
$ MOUNT/SHARE DMA3:  PUBS_BACK
```

This MOUNT command indicates that other users can access the volume PUBS_BACK. Each user who wants to access the volume must also issue a MOUNT command with the /SHARE qualifier. Note that if the device was allocated before the MOUNT command was issued, the /SHARE qualifier deallocates the device.

Access to the volume is restricted according to the protection code on the volume. The system identifies the volume by its volume label, rather than by the device on which it is mounted. Other users who want to access the volume do not need to know the physical name of the device, but only the generic device type and volume label.

For example, another user who wants to use the volume mounted in the example above would issue the MOUNT command as follows:

```
$ MOUNT/SHARE  DM:  PUBS_BACK PUBS
```

Thereafter, the device can be referred to by the logical name PUBS; the sharer does not need to know the name of the physical device.

Because the system uses volume labels to identify shared volumes, two volumes that have the same label cannot be mounted with the /SHARE (or /SYSTEM or /GROUP) qualifier at the same time.

3.4.1.3 **Disk Quotas** - Frequently it is important to limit the amount of disk space certain users consume. The DISKQUOTA utility provides the system manager this capability. Users who have read access (R) to the quota file can use the SHOW QUOTA command to determine how much disk space any user has been allocated. Other users who lack this authorization can at least determine what their own allotments are. If the need arises to disable this limitation, the MOUNT command offers the /QUOTA and /NOQUOTA qualifiers to manage the checking on a per volume basis.

A user can have a quota for any volume on the system. In some cases, the user is permitted a certain authorized limit plus an overdraft limit. Generally, only the authorized limit applies. However, certain system programs, such as editors, can employ the overdraft feature when the authorized limit is exceeded.

```
$ SHOW QUOTA
User [360,010] has 3020 blocks used, 6980 available,
of 10000 authorized and permitted overdraft of 500 blocks on DSK1
```

This SHOW QUOTA command identifies the disk quota authorized to the current user on the current default disk DSK1 is 10000 blocks, of which 3020 blocks are used. Thus, user [360,010] has 6980 blocks available, and under some circumstances may even be allowed an additional 500 blocks from the overdraft quota.

If you should run out of disk space during the creation of a file, you receive an error message. If you find you cannot obtain sufficient space by purging or deleting unnecessary files, you may need to contact the system manager to increase your disk quota. The VAX/VMS System Manager's Guide describes disk quotas and how to use the DISKQUOTA utility.

3.4.2 **Disk Volume Sets**

Using VAX/VMS, you can bind two or more disk volumes into a volume set. A volume set has a single directory structure; the MFD (master file directory) for the entire volume set exists on the first volume in the set, called the root volume. Each volume in the set is identified by a relative volume number in the set, where the root volume is always relative volume 1.

When a volume set is online and mounted, all files and directories in the set can be accessed by specifying the device name of the device on which the root volume is mounted, or the logical name assigned to the volume set when it was mounted.

The binding of volumes into volume sets allows the system manager to extend the space available for user's files adding volumes to the volume set, rather than defining new volumes with additional directories.

To create a volume set, you use the MOUNT command with the /BIND qualifier. The /BIND qualifier identifies a set by giving it a volume set name, which applies to all volumes in the set, and it identifies the root volume and creates the directory structure for the volume.

Once a volume set has been created:

- All users who have directories and files on the set can access their files either by referring to the physical device name of the device on which the root volume is mounted or by referring to a logical name established for the volume set.

- When users create files on a volume set, the file system allocates space for the files anywhere on the set, wherever there is the most room.

- When existing files on any volume are extended, extension occurs on the same volume, unless the volume is physically full.

- New volumes can be added to a volume set whenever additional space is required.

For example, all disk volumes that are mounted on a daily basis can be bound into a volume set. Since this set contains all user file directories, users do not need to specify device names in file specifications to access files that would be on other volumes. In fact, the physical location of a file is transparent to all users of the system.

The next sections describe the procedures for creating and mounting volume sets, and contain additional notes on volume sets.

3.4.2.1 **Creating a Volume Set** - You can create a volume set from new, freshly initialized volumes or you can create a volume set by extending an existing volume that already contains a directory structure and files.

No special privileges are required to create or use volume sets; however, you must have write access to the index file on all volumes that you are attempting to bind into a volume set. In general, this means that you must have a system UIC, have the user privilege SYSPRV, or be the owner of the volumes.

3.4.2.2 **Creating a Disk Volume Set from New Volumes** - This procedure assumes that none of the volumes to be bound into a volume set contains files or data.

1. Allocate the necessary devices and physically mount the volumes.

2. Initialize each volume in the set. For example:

   ```
   $ INITIALIZE DB1:  PAYVOL1
   $ INITIALIZE DB2:  PAYVOL2
   $ INITIALIZE DB3:  PAYVOL3
   ```

   When you initialize volumes for a volume set, you can also use other qualifiers on the INITIALIZE command to define the volume ownership and protection. Although not required, it is recommended that all volumes in a set have the same protection and the same owner.

3. Use the MOUNT/BIND command to create the volume set. For example:

   ```
   $ MOUNT/BIND=MASTER_SET -
   $_DB1:, DB2:, DB3:  PAYVOL1, PAYVOL2, PAYVOL3
   ```

This MOUNT/BIND command defines the volume set name, MASTER_SET, and defines the relative volume numbers of the volumes PAYVOL1, PAYVOL2, and PAYVOL3.

A volume set name can have from 1 through 12 alphanumeric characters; the volume set name must be different from all volume labels within the set and all labels in the set must be unique.

The order of the device names corresponds to the volume labels specified: PAYVOL1 must be physically mounted on DB1, PAYVOL2 on DB2, and PAYVOL3 on DB3.

PAYVOL1, because it is listed first in the list of labels, becomes the root volume of the set. Its master file directory (MFD) contains the directory structure for the entire set.

Note that the MOUNT/BIND command creates the volume set and mounts the volumes. When this command completes successfully, all volumes in the set are ready for use: user file directories can now be created.

3.4.2.3 **Creating a Disk Volume Set from an Existing Volume** - The following example assumes that the volume USERFILES already contains a directory structure and files and that the volume is currently mounted on the device DB1.

```
$ MOUNT/SYSTEM/BIND=USERS -
$_DB1:, DB2:    USERFILES, USERFILES2
```

The initial volume USERFILES must be specified first: it becomes the root volume of the set. When you create a volume set from an existing volume, you must specify that volume first because the file system must build on the existing directory structure.

Note that if you attempt to create a volume set from two or more volumes that already contain files and data, the file system does not issue an error message when you issue the MOUNT/BIND command. However, the volumes are unusable as a volume set because the directory structures are not properly bound.

3.4.2.4 **Mounting a Disk Volume Set** - When you mount an existing volume set, you must specify the names of the devices on which the volumes are mounted and the volume labels in a corresponding order. The MOUNT command verifies the label on each device/volume pair specified in the list. For example:

```
$ MOUNT/SHARE DB1:, DB2:, DB3:, -
$_PAYVOL1, PAYVOL2, PAYVOL3
```

You can also issue separate MOUNT commands for each device and volume in the set. For example:

```
$ MOUNT/SHARE DB1:  PAYVOL1
$ MOUNT/SHARE DB2:  PAYVOL2
$ MOUNT/SHARE DB3:  PAYVOL3
```

The effect of these commands is the same as that of the previous MOUNT command example: the three volumes in the set are mounted.

Note that the file system does not require that all volumes in a volume set be mounted. If a user attempts to access a file in a volume set and the volume is not currently mounted or the root volume is not mounted, the error status DEVNOTMOUNT is returned. Mount the remainder of the volume set and try again.

<u>Volume Status</u> - When you mount a volume set by mounting volumes individually, you must ensure that the MOUNT commands define the volume in the same way. For example, if one or more volumes are mounted for sharing with the /SHARE qualifier initially, all subsequent volumes must also be mounted with the /SHARE qualifier.

For each user the file system maintains the names of volume sets in two lists corresponding to the possible statuses. In each list, volume set names must be unique. There is one list consisting of the names of all volume sets that are mounted privately and another list consisting of the names of all volume sets that are mounted as shareable. A volume's desired status is defined on the MOUNT command. The possible statuses and the corresponding qualifiers that define them are:

Status	**Qualifier**
Private	/NOSHARE
Shared	/SHARE
Group shared	/GROUP
System	/SYSTEM

Thus, if a volume set is mounted with the /SYSTEM qualifier and subsequently a request is made to bind another volume to the set and the /SYSTEM qualifier is omitted, the volume set name is placed in the list corresponding to volume sets mounted privately. Assuming that no volume set of the specified name is mounted with the /NOSHARE qualifier, the new volume will become relative volume 1 of a new volume set, because the volume set was not found. To correctly bind it into the existing volume set, you must dismount the volume, reinitialize it, and then remount it.

Logical Names - When you mount a volume set, you can specify a logical name for the set or you can allow the MOUNT command to make default logical name assignments. A logical name for a volume set can be used to refer to all volumes in the set. For example:

```
$ MOUNT/SHARE DB1:, DB2:, DB3:  -
$_PAYVOL1, PAYVOL2, PAYVOL3   PAY
```

This MOUNT command mounts three volumes in a volume set and assigns the logical name PAY to the set. Users who are sharing this volume set can use the logical name PAY in place of the device name in file specifications to refer to the set, as follows:

```
$ PRINT PAY:[WEEKLY.JAN0878]EMPLOY.LIS
```

If you do not specify a logical name, the MOUNT command assigns the default logical name DISK$volume-set-name to the root volume, that is, to the device on which the root volume is mounted. If the root volume is not mounted, no logical name assignment is made. Each volume in the set is also assigned a default logical name of the format DISK$volume-label. However, since there is normally no need to refer to individual volumes in a volume set, except for maintenance purposes, these names are rarely used.

The MOUNT command places the logical name for the volume set and for individual volumes in different logical name tables based on the status of the set:

Status	**Logical Name Table**
Group	Group
Private	Process
Shared	Process
System	System

The user privileges GRPNAM and SYSNAM are required to place names in the group and system logical name tables, respectively. Hence, these privileges are required to mount a volume set in either group or system status.

3.4.2.5 **Adding Volumes to a Disk Volume Set** - You can add volumes to an existing volume set at any time. The maximum number of volumes in a set is 255.

The following procedure assumes that the volume set named MASTER_PAY is online and mounted and has volumes named PAYVOL1, PAYVOL2, and PAYVOL3:

```
$ INITIALIZE DB4:  PAYVOL4
$ MOUNT/BIND=MASTER_PAY DB4:  PAYVOL4
```

This MOUNT command binds the volume PAYVOL4 with the existing volume set and makes the volume ready and available for use. Note that if the volume set MASTER_PAY was mounted in a system, group, or shared status, the MOUNT/BIND command that adds a volume to the set must also specify the appropriate qualifier.

When you add a volume to an existing set, the only volume in the set that must be mounted is the root volume, relative volume 1. None of the other volumes need be mounted.

You can also add a volume to a set at the same time that you are mounting the set. The following procedure assumes an existing volume set named MASTER_SET with volumes named PAYVOL1, PAYVOL2, and PAYVOL3:

```
$ INITIALIZE DB4:  PAYVOL4
$ MOUNT/BIND=MASTER_SET DB1:, DB2:, DB3:, DB4:  -
$_PAYVOL1, PAYVOL2, PAYVOL3, PAYVOL4/SYSTEM
```

Note, in the above example of the MOUNT command, that the first device/volume pair listed in the command is the root volume of the set. When you add a volume to a set while mounting the set, you must list the root volume first.

3.4.2.6 **Dismounting Disk Volume Sets** - To dismount an entire volume set, use the DISMOUNT command specifying the name of any device on which a volume of the set is mounted. For example:

```
$ DISMOUNT DB1:
```

By default, the DISMOUNT command dismounts all volumes in the set that are currently mounted.

You can use the /UNIT qualifier on the DISMOUNT command to request that only the volume on the specified device be dismounted, if necessary.

3.4.3 Using Tapes

The default format for reading and writing tapes is based on the ANSI X3.27-1978 Magnetic Tape Labels and File Structure for Information Interchange standard.

The following examples show how to allocate, initialize, and use a tape to back up your disk files. The procedures are similar to those outlined above for using disk volumes. However, tapes are sequential access devices and do not have directories.

```
$ ALLOCATE MT:
  _MTA2:  ALLOCATED
```

This ALLOCATE command requests the allocation of a tape device whose name begins with MT; the response indicates that unit 2 on controller A was available and is now allocated to you. You can now physically load the tape on the drive. Next, initialize the tape:

```
$ INITIALIZE  MTA2:  GMB001 -
$_/PROTECTION=(GROUP:R,WORLD)
```

The INITIALIZE command specifies the device name (MTA2) and the volume label for the tape volume (GMB001). The /PROTECTION qualifier defines a protection code restricting group access to read and allowing no access to the world. You can now use the MOUNT command to mount the volume and write files to it:

```
$ MOUNT  MTA2:  GMB001
%MOUNT-I-MOUNTED, GMB001 mounted on _MTA2:
$ COPY *.*  MTA2:
```

The MOUNT command specifies the device name and volume label of the volume on the device. The COPY command copies the highest versions of all files in your default directory onto the tape. The file names and file types of the output files default to the same file names and file types as the input files.

To verify that the files were successfully copied, you can use the DIRECTORY command:

```
$ DIRECTORY MTA2:
```

The DIRECTORY command lists the file names and file types of all files on the tape.

When you have finished using the tape, dismount it and deallocate it, as shown below:

```
$ DISMOUNT MTA2:
$ DEALLOCATE MTA2:
```

If you do not dismount and deallocate the tape, the system does so automatically when you log out.

3.4.3.1 **Reading and Writing Tape Files** - A magnetic tape is a sequential device. With DCL commands, you can write new files only at the end of existing data on the tape; you cannot delete files from a tape. You can, however, append data at the end of an existing file on a tape; all files that follow the appended file are overwritten. The APPEND command will not, however, overwrite a file that has not expired. If you want to overwrite a tape completely, you must re-initialize it.

When you want to copy files from an existing tape, you can selectively copy files from the tape by specifying the file name and file type of the file you want to copy, as shown below.

```
$ MOUNT MTB2:  GMB001
$ COPY MTB2:ASTRO.SRC  ASTRO.OLD
  .
  .
  .

$ COPY MTB2:ASTRO.FOR  ASTRO2.FOR
$ DISMOUNT MTB2:
```

The COPY commands above copy specific files from the tape. After each copy request, the COPY command leaves the tape positioned at the end of the file it has just copied. When it looks for the next file, it continues searching until the end of the tape; if it does not find the file, it rewinds the tape and searches until it either locates the file or returns to the point on the tape at which it started.

3.4.3.2 **Version Numbers for Tape Files** - Files that you write onto tape with DCL commands are written in sequential order and have version numbers of 0, by default. The only exception to this is the COPY command, which gives the copy the same version number as the original, by default. If you want to read a file from a tape and you do not specify a file version number, the command locates the next file with that file name and file type that is physically on the tape.

3.4.3.3 **Writing Tapes with Compatibility Mode Programs** - You can write to tapes using programs and utilities that execute in RSX-11M compatibility mode (for example, some editors and compilers). However, you should be aware of the following:

- All compatibility mode programs (except PIP) limit I/O buffer space to 512-byte blocks. However, the default block size for tapes on VAX/VMS is 2048 bytes. For magnetic tapes that will be written by compatibility mode programs you should specify /BLOCK=512 when you mount the tape.
- If a compatibility mode program or utility encounters an error creating a file on a magnetic tape, the tape will likely be left improperly written. It will have a header label set for the file being created, but no file or volume trailer label set. Attempting to create more files on this tape or listing files on the tape will cause the tape drive to read past the header label set into uninitialized tape. This causes I/O errors, runaway tape, and other random behavior. The only recovery from this situation is to copy the good files from the tape and then reinitialize it. (Because you name each file you copy, you do not read past the header label into uninitialized tape.)

3.4.4 Multivolume Tape Sets

The VAX/VMS operating system allows you to create and access files that span more than one physical tape volume. Each volume in a multivolume set has a unique volume label and a relative volume number within the set.

Processing of multivolume files requires the attendance of an operator or user to respond to requests to switch volumes; when a command or program attempts to read or write beyond the end of a tape, the system automatically sends a message to the system operator's console (or to a terminal designated as an operator's terminal). The message indicates:

- The device name
- The relative volume number of the next volume in the volume set
- The label, if known

Before processing can continue, the operator must physically mount the volume, place the device online, and type a reply to the message. If no operator is available, you can load the volume yourself. You can login on the operator's console or another operator's terminal and reply to the message yourself. You do not need the operator user privilege (OPER) to do this.

When you issue the MOUNT command to begin processing a multivolume file, you can specify the labels on each of the volumes in the MOUNT command. For example, after physically loading the first volume on the device MTA0, you can issue the MOUNT command as follows:

```
$ MOUNT MT:  GMB001, GMB002, GMB003
```

The MOUNT command verifies the label on the first volume and returns a message indicating successful completion if the volume label is correct.

Subsequently, when the tape reaches end-of-tape, the system automatically rewinds the tape and sends a request to the system operator to mount the volume labeled GMB002. The messages the operator receives typically look like the following:

```
Opcom, 02:18:48.01, GEOFF     Accnt=TEXTPROC  Reply-ID=131084
Opcom, MOUNT relative volume 2 (GMB002) on MTA0:
```

After loading the volume and readying the device, the operator types the command:

```
$ REPLY/TO=131084
```

The REPLY command requires the operator user privilege (OPER) and is described in the VAX/VMS Operator's Guide.

The system verifies the reply-ID and then verifies the volume. If the correct volume is mounted, processing continues.

At the end of that tape, the system sends a message to mount the tape labeled GMB003. No more explicit MOUNT commands are required.

3.4.4.1 **Creating a Multivolume Tape Set** - When you initially create a file that spans tape volumes, you may not know that multiple volumes are required. If, while you are writing to the tape, the tape reaches end-of-tape, the system suspends processing to notify the operator that a new volume is required. In this case, the system does not know the volume label. The operator must mount a volume and enter the volume label on the REPLY command as shown below.

```
$ REPLY/TO=196620 "GMB004"
```

If the tape is a new tape, the operator can request that it be initialized by specifying /INITIALIZE following the label. For example:

```
$ REPLY/TO=196620   "GMB004/INITIALIZE"
```

The system performs normal protection and expiration checks before initializing the volume. To override the checking for protection information on tapes that have been processed by a verifying machine, the operator can specify /BLANK. For example:

```
$ REPLY/TO=196620 "GMB004/BLANK"
```

3.4.4.2 Using Multiple Tape Drives

You can overlap the mounting of volumes in a multivolume set by specifying more than one drive in the MOUNT command. For example:

```
     $ MOUNT  MTA0:,MTA1:  GMB001, GMB002, GMB003
```

The MOUNT command verifies the volume on MTA0. If the volume GMB002 is located on the device MTA1 when the end-of-volume is reached on GMB001, processing continues. However, when the end-of-volume occurs on GMB002, the first volume is rewound on MTA0 and the system sends a message to the operator to MOUNT the third volume on MTA0.

3.5 ACCESSING DEVICES IN BATCH JOBS

You can write command procedures to mount a volume from a batch job. By using logical names to refer to devices and files, you can use the same command procedures without modification each time you want to access a volume.

For example, if you use the same RK07 disk pack to back up your files on a weekly basis, you can submit as a batch job a command procedure like the following:

```
$ TRY:
$ ALLOCATE  DM:  RK
$ IF  $STATUS  THEN  GOTO  OKAY
$ WAIT  00:05
$ GOTO  TRY
$ OKAY:
$ REQUEST/REPLY/TO=DISKS  -
"PLEASE MOUNT  RK07 BACK_UP_GMB ON ''F$LOGICAL("RK")' "
$ MOUNT  RK  BACK_UP_GMB
$ COPY/REPLACE  *.*  RK:*.*
$ DIRECTORY/FULL/OUTPUT=BACKUP.LOG  RK:
$ DISMOUNT RK
$ PRINT  BACKUP.LOG
$ DEALLOCATE  RK
$ REQUEST/TO=DISKS    "All done, thanks..."
```

In this job, the procedure places itself in a wait state for five minutes (if no RK07 is available) and loops to repeat the request. When the ALLOCATE command completes successfully, a message is sent to the operator.

The logical name, RK, assigned on the ALLOCATE command is used in all subsequent commands. The lexical function F$LOGICAL is used in the REQUEST command; this function translates the logical name RK and substitutes the equivalence name in the message displayed at the operator's console.

When the REQUEST command notifies the operator to mount the correct volume, the job waits until the operator responds before continuing.

For more information on how to create and execute command procedures, see the VAX/VMS Guide to Using Command Procedures.

CHAPTER 4

PROGRAMMING WITH VAX/VMS

The VAX/VMS operating system provides concurrent time-shared multiprogramming and batch job processing: many users can be logged in at terminals to create and test new programs and applications interactively at the same time that production applications and real-time process control applications are running.

This chapter describes:

- Commands for program development
- Debugging programs
- Exit handlers and condition handlers
- Process concepts

Table 1-5 in Section 1.10, "Summary of VAX/VMS DCL Commands" lists the commands described in this chapter. For details on the parameters and qualifiers for any of these commands, see the command descriptions in Part II.

4.1 COMMANDS FOR PROGRAM DEVELOPMENT

Figure 4-1 illustrates the steps required to create and execute programs in VAX/VMS.

The following example illustrates the DCL commands you could use to create, compile, link, and execute a FORTRAN program named AVERAGE:

```
$ EDIT/SOS AVERAGE.FOR
Input:DBA1:[MALCOLM]AVERAGE.FOR;1
00100
   .
   .
   .

   (input source statements)
   .
   .
   .
ESC

*e
$ FORTRAN AVERAGE
$ LINK AVERAGE
$ RUN AVERAGE
```

Use the *editor* to create a disk file containing your source program statements. Specify the name of this file when you invoke the compiler or assembler.

Commands invoke language processors that check syntax, create object modules, and if requested, generate program listings.

If a processor signals any errors, use the editor to correct the source program.

The *linker* searches the system libraries to resolve references in the object module and create an executable image. Optionally, you can specify private libraries to search, and request the linker to create a storage map of your program.

The linker issues diagnostic messages if an object module refers to subroutines or symbols that are not available or undefined. If the linker cannot locate a subroutine, you must reissue the LINK command specifying the modules or libraries to include. If a symbol is undefined, you may need to correct the source program.

The *RUN* command executes a program image. While your program is running, the system may detect errors and issue messages. To determine if your program is error-free, check its output.

If there is a bug in your program, determine the cause of the error and correct the source program.

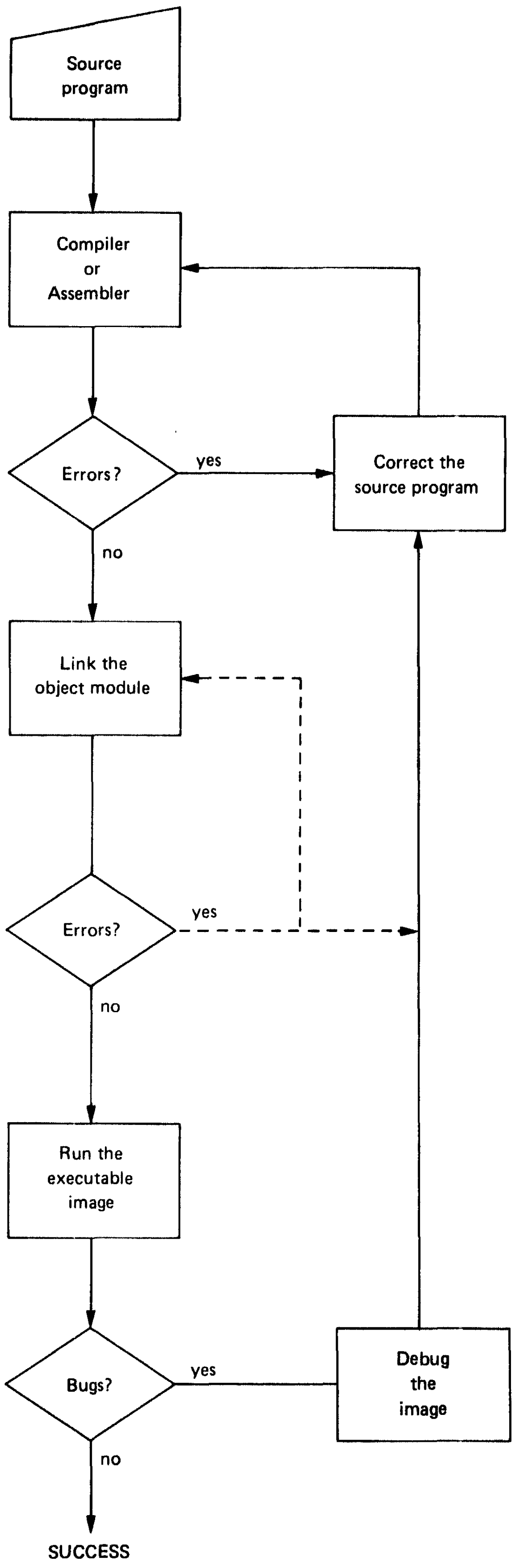


Figure 4-1 Steps in Program Development

Note that the EDIT command invokes the SOS editor. SOS prompts for input lines until you use ESC to terminate input. Then, the E (Exit) command requests SOS to write the input data onto disk.

The FORTRAN command invokes the VAX-11 FORTRAN Compiler to compile the source statements.

The commands EDIT, FORTRAN, LINK, and RUN, are shown in their simplest forms, without qualifiers. By giving the source file a file type of FOR in the file creation, you can allow the file types of the input and output files to assume the defaults for the FORTRAN, LINK, and RUN commands.

There is a command to invoke each language processor, and each language processor assumes a default file type for the input file. The language processors, the commands to invoke them, and the default input file types used by each are summarized in Table 4-1.

Table 4-1
Default Input File Types for Language Processors

Command	Language Processor	File Type
BASIC	VAX-11 BASIC[1]	BAS
BASIC/RSX11	PDP-11 BASIC-PLUS-2/VAX[1]	B2S
BLISS	VAX-11 BLISS-32[1]	B32 or BLI
COBOL/C74	VAX-11 COBOL-74[1]	COB
COBOL/RSX11	PDP-11 COBOL-74/VAX[1]	CBL
CORAL	VAX-11 CORAL 66[1]	COR
FORTRAN	VAX-11 FORTRAN[1]	FOR
MACRO	VAX-11 MACRO	MAR
MACRO/RSX11	PDP-11 MACRO	MAC
PASCAL	VAX-11 PASCAL[1]	PAS

1. Available under separate license.

You can find a description of each of these commands and lists of the valid qualifiers in Part II of this manual.

The next few sections discuss DCL commands and system programs that can help you develop, test, and maintain your programs.

4.1.1 **Program Libraries**

The LIBRARY command creates and maintains object macro, text, or help libraries. A library is a file that contains its own index of the entries it contains.

4.1.1.1 **Object Module Libraries** - Object module libraries are convenient for storing routines that are called frequently by many programs. For example, if you have compiled the object modules named TIMER, CALC, and SWITCH, you could create a library named COMMON.OLB using the LIBRARY command as follows:

```
$ LIBRARY/CREATE  COMMON  TIMER,CALC,SWITCH
```

When you issue a LINK command to link an object module that calls any of these routines, you can specify the library as a linker input file using the /LIBRARY qualifier:

```
$ LINK  TESTPROG,COMMON/LIBRARY
```

When the linker links the module TESTPROG, it then searches the library COMMON.OLB if it encounters any undefined external references. Alternatively, you can make COMMON.OLB a user-defined default library, as described in the VAX-11 Linker Reference Manual.

The system object module library, STARLET.OLB, contains system routines that are called frequently. The linker automatically searches this library after searching any private libraries you specify for undefined external references.

4.1.1.2 **Macro Libraries** - If you are a MACRO programmer, you can also use the LIBRARY command to catalog macros that you use frequently. For example, to create a macro library named LOCALMAC.MLB from the macros contained in the files DESCRIPTOR.MAR, TRANSLATE.MAR, and RANDOM.MAR, issue the command:

```
$ LIBRARY/CREATE/MACRO  LOCALMAC  DESCRIPTOR,TRANSLATE,RANDOM
```

To assemble a program that invokes any of these macros, specify the name of the library on the MACRO command with the /LIBRARY qualifier, as shown below:

```
$ MACRO RUNTEST+LOCALMAC/LIBRARY
```

This MACRO command indicates that there are two input files: RUNTEST.MAR, the source file, and LOCALMAC.MLB, a library.

The system library STARLET.MLB contains system macros. The assembler automatically searches this library to locate macro definitions after searching any private libraries you specify, as in the above example.

4.1.1.3 **Help Libraries** - Help messages can provide specific information about a program to an interactive user. Once your help messages are stored as modules in help libraries, your programs can access the help modules by calling the appropriate Librarian routine as described in the VAX-11 Utilities Reference Manual. Use the DCL LIBRARY command to maintain the help libraries.

4.1.1.4 **Text Libraries** - Text libraries contain any sequential record files that you want to retrieve as data for your program. For example, program comments can be stored in text libraries. The LIBRARY command maintains text libraries. Your programs can retrieve text from text libraries by calling the appropriate Librarian routine as described in the VAX-11 Utilities Reference Manual.

4.1.2 **Controlling Program Updates and Modifications**

VAX/VMS provides several commands and programs you can use to track and control updates that you make to your programs.

4.1.2.1 **Updating Source Programs** - To update or modify a source program, you can use the interactive editors, SOS or EDT. Because SOS, by default, creates a new file (with a higher version number) each time you edit a file, you can keep previous versions of a file for back-up. SOS does not, however, provide you with a record of the changes that you have made. EDT offers this capability through its /JOURNAL and /RECOVER qualifiers. VAX/VMS also provides two batch-oriented editors, SLP and SUMSLP. With SLP, you can insert, delete, or replace lines in a file and create a new file incorporating your changes. SLP also creates a record of all the modifications that you made. With SUMSLP you can apply edit commands in multiple files against a single input file.

SOS, EDT, SLP, and SUMSLP are invoked with the EDIT command. The SOS editor is described in detail in the VAX/VMS Text Editing Reference Manual. The SLP and SUMSLP editors are described in the VAX-11 Utilities Reference Manual. The EDT editor is described in the VAX-11 EDT Editor Reference Manual.

4.1.2.2 **Comparing Versions of Files** - The DIFFERENCES command invokes a file comparison program that compares the contents of two files to determine what differences, if any, exist between them. DIFFERENCES creates an output file that summarizes the differences.

The DIFFERENCES command is described in Part II.

4.2 DEBUGGING

The VAX-11 Symbolic Debugger is an interactive debugging program. It has an extensive set of commands that allow you to examine and modify a program in memory while it is executing.

The debugger uses three sets of information:

- Local symbol table information
- Global symbol information
- Traceback information

When you use DCL commands to compile or assemble and link a program, you can control what information, if any, you want to include in the image.

If you request local symbol table information, you can refer to actual variable names and statement labels when you issue DEBUG commands. If you request traceback information, the debugger can trace the call stack when an error occurs during image execution.

By default, many of the language compilers such as VAX-11 FORTRAN and the VAX-11 MACRO assembler generate traceback information, but not local or global symbol information, in the object module. The linker, also by default, includes the traceback information in the image file so that you receive a symbolic traceback when an error occurs.

4.2.1 Symbolic Debugging

If you want to use the complete symbolic capabilities of the debugger, you must request the debugger when you compile and when you link a program. The following example shows the commands to compile a FORTRAN program that includes all symbols in the image and automatically invokes the debugger when run:

```
$ FORTRAN/DEBUG/NOOPTIMIZE  PRECIP
$ LINK/DEBUG  PRECIP
$ RUN PRECIP

                VAX-11 DEBUG V2.0

%DEBUG-I-INITIAL, language is FORTRAN, module set to 'PRECIP'
DBG>
```

The FORTRAN command above requests that the debugger symbol table and traceback information be included in the object module (the /NOOPTIMIZE qualifier ensures a one-to-one correspondence between the source program statements and the machine code in the object module).

The LINK command above requests the automatic inclusion of the debugger and local and global symbol definitions in the image.

When you issue the RUN command to execute an image linked with the debugger, the debugger receives control, identifies itself, and prompts for you to begin entering DEBUG commands. If you do not want the debugger to prompt when you execute an image, issue the RUN command as follows:

```
$ RUN/NODEBUG PRECIP
```

In this case, the debugger does not prompt; however, you can interrupt the program and issue the DEBUG command after a CTRL/Y.

For complete descriptions of the DEBUG commands and considerations for using the debugger, see the VAX-11 Symbolic Debugger Reference Manual. You should also consult the user's guide for your programming language for information specific to debugging programs in that language.

Note that symbol table information and, to a lesser extent, traceback information increase the size of an object module and the executable image. When you have debugged a program, you can recompile or reassemble without the symbol table information, retaining traceback information in the event of unexpected errors in the future.

You can also exclude traceback information from modules that you catalog in object module libraries. Otherwise, the traceback information is included in all modules that link with the library module.

4.2.2 Debugging with Virtual Addresses

You can debug an image that was not compiled and linked with the debugger symbol table. The /DEBUG qualifier on the RUN command requests the debugger at run time. For example:

```
$ RUN/DEBUG  ORION
```

To specify memory locations for the debugger, you must have both a machine code listing of the object module and a full map from the linker. The map gives the virtual address (in hexadecimal) of each module, global symbol, and program section (PSECT) in the image.

If you do not link or run an image with the debugger, you can debug a program using virtual addresses with the DCL commands EXAMINE and DEPOSIT. The EXAMINE command displays the current contents of a location or range of locations; the DEPOSIT command modifies a location. These commands provide a useful, but limited, debugging capability when you need to debug a program that cannot be run with the debugger. The DEPOSIT and EXAMINE commands are described in detail in Part II.

4.2.3 Interrupting Program Execution

When you use the RUN command to execute an image interactively, you cannot execute any other images or DCL commands until the image exits. If you enter a command line, the system saves the line in the terminal type-ahead buffer, but does not process it until the image exits.

If you need to interrupt an image while it is executing, press CTRL/C or CTRL/Y. Generally, the effect of both of these CTRL key sequences is the same: the image is interrupted (but unchanged), the type-ahead buffer is purged, and the command interpreter receives control.

Note: Some system or application programs may contain special routines coded to intercept a CTRL/C. If you use CTRL/C to interrupt these programs, the CTRL/C handling routine receives control, rather than the command interpreter. Use CTRL/Y to bypass a CTRL/C handling routine. If a command or program does not have a CTRL/C handling routine, then CTRL/C has the same effect as CTRL/Y and echoes as ^Y. For information on coding CTRL/C handling routines, see the VAX/VMS I/O User's Guide.

The following example shows CTRL/C or CTRL/Y being pressed to interrupt a program that is looping:

```
$ RUN NAMETST
ENTER YOUR NAME:
ENTER YOUR NAME:
ENTER YOUR NAME:
^Y
$
```

The dollar sign ($) prompt indicates that you can enter a DCL command. After you have interrupted an image (or a DCL command) with CTRL/Y, you can:

- Enter the EXIT command. This causes orderly termination of the interrupted image
- Immediately abort the image, by entering the STOP command
- Issue the CONTINUE command to resume execution of the image from the point of interruption.
- Issue the DEBUG command, if the image was linked without specifying either /NOTRACEBACK or /NODEBUG or run with the /DEBUG qualifier. The DEBUG command gives control to the debugger.

NOTE

If the image you interrupt is privileged, that is installed with privilege, it terminates immediately, just as if you had entered an EXIT command. You can neither continue nor debug the image. The creation of privileged images with the INSTALL Utility is described in the VAX/VMS System Manager's Guide.

You can also issue any other DCL command. Most DCL commands you enter at the CTRL/Y level have the same effect as the RUN command; that is, the current image is forced to exit before the command can be executed. However, the commands in Table 4-2 are performed within the command interpreter and thus do not cause the current image to exit.

Table 4-2
Commands Performed Within the Command Interpreter

=	EXAMINE	SHOW DAYTIME
ALLOCATE	GOTO	SHOW DEFAULT
ASSIGN	IF	SHOW QUOTA
CLOSE	INQUIRE	SHOW PROTECTION
CONTINUE	OPEN	SHOW STATUS
DEALLOCATE	READ	SHOW SYMBOL
DEASSIGN	SET DEFAULT	SHOW TIME
DEFINE	SET PROTECTION/DEFAULT	SHOW TRANSLATION
DELETE/SYMBOL	SET VERIFY	WAIT
DEPOSIT	SET UIC[1]	WRITE

1. This command is described in the VAX/VMS Operator's Guide.

For example, you could interrupt any command or program (except a privileged image), issue the SHOW TIME command, and then continue execution of the image, as follows:

```
$ RUN GRADES
^Y
$ SHOW TIME
  07-JUN-1978 10:54
$ CONTINUE
```

Note that you can also use CTRL/O, CTRL/S, and CTRL/Q to suppress, interrupt, or continue terminal output from a program image, just as you can for the output of a DCL command. These control key functions do not affect the program image.

4.3 EXIT HANDLERS AND CONDITION HANDLERS

Programs that execute in native mode can take advantage of operating system services that allow an image to respond to special situations.

4.3.1 Exit Handlers

An exit handler is a special routine that receives control when the image exits. The exit handler can determine whether the image is exiting normally or as the result of an error condition.

If you have used the RUN command to execute an image that has an exit handler, and you interrupt the image with CTRL/Y, you can control whether the exit handler is actually given control. If you issue the STOP command, the exit handler is not executed. If, on the other hand, you issue the EXIT command or another DCL command to execute another image, the exit handler in the interrupted image is allowed to execute before the specified command is performed.

For more information on exit handlers, see the VAX/VMS System Services Reference Manual.

4.3.2 Exception Conditions

The system interrupts image execution when the image causes or encounters particular situations, called exceptions. Some examples of exceptions are:

- Arithmetic overflow or underflow
- A memory access violation
- An invalid operation code
- An invalid argument list to the math library

When an exception occurs, the system searches for a routine that can respond to the condition; these routines, called condition handlers, can be declared from user programs. If no user-declared condition handlers are located, or if a user-declared condition handler cannot respond to a particular situation, the system gives control to a default handler. This handler terminates the image; obtains as much information as it can about the exception condition, including any available traceback information; and summarizes the information for you.

Further discussion of condition handling appears in the VAX/VMS System Services Reference Manual, the VAX11 Architecture Handbook, the VAX11/780 Hardware Handbook, and the VAX-11 Run-Time Library Reference Manual.

4.4 PROCESS CONCEPTS

The executable program image created by the linker executes within the context of the process created for you at login. This process can execute, serially, many different images during your terminal session. When you issue the LOGOUT command to end your terminal session, the system deletes the process.

The VAX/VMS operating system manages all users' requests to execute images in terms of the processes issuing the requests. The system provides each process with (among other things), a unique process identification number (PID), a character-string process name, and a distinct environment. Figure 4-2 illustrates the relationship of a terminal user to the process and the images executed in the process.

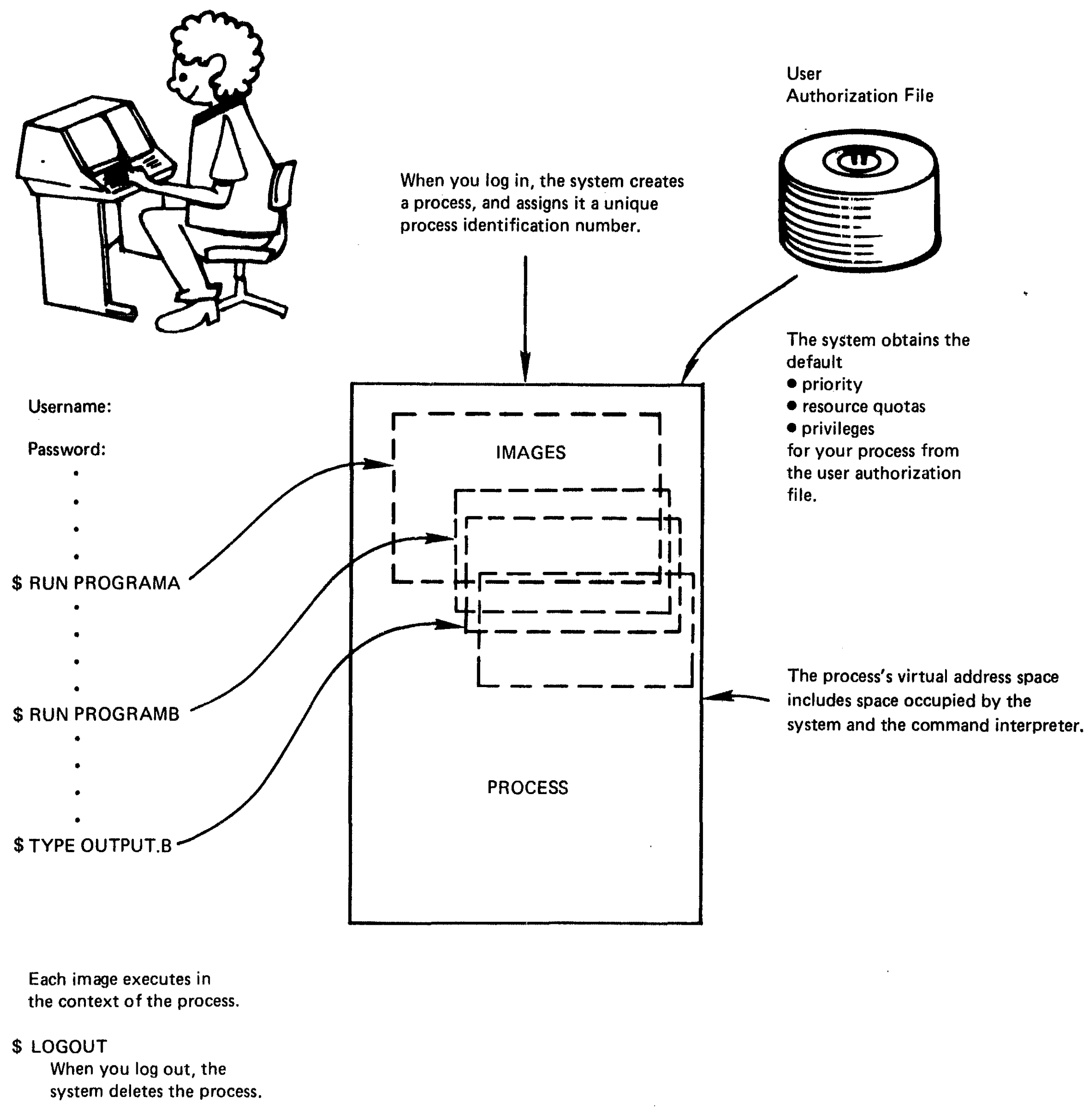

User Authorization File
When you log in, the system creates a process, and assigns it a unique process identification number.
The system obtains the default
● priority
● resource quotas
● privileges
for your process from the user authorization file.
Username:
Password:
$ RUN PROGRAMA
$ RUN PROGRAMB
$ TYPE OUTPUT.B
IMAGES
PROCESS
The process's virtual address space includes space occupied by the system and the command interpreter.
Each image executes in the context of the process.
$ LOGOUT
When you log out, the system deletes the process.

Figure 4-2 An Interactive Process

4.4.1 Priorities, Privileges, and Quotas

The characteristics of an individual process -- that is, the process's context -- determine what the user may do or be granted access to, and how much of the various resources may be consumed. Most of these characteristics are obtained from the user authorization file at login time.

Some examples include:

- The base execution priority given to the images that the process executes. In general, communications and process control applications are given higher priorities than batch jobs and interactive users.

- Resource quotas that limit or restrict the frequency or amount of a particular system resource any image can use. For example, an open file quota limits the number of files a process can have open at any one time.
- User privileges to perform certain functions or to call specific system services. For example, the ability to place names in the group or system logical name tables is controlled by a privilege.

The base priority, resource quotas, and privileges granted to general users are adequate for time-sharing program development requirements. In fact, many of the DCL commands you execute perform privileged functions on your behalf, so you do not require the privilege. Note, however, that for some commands you must have a user privilege to use a particular qualifier. These restrictions are noted in the command descriptions, as appropriate.

You can determine the current privileges and quotas available to your process by issuing the following command:

```
$ SHOW PROCESS/PRIVILEGES/QUOTAS
```

Tables 1-7 and 1-8 list the privileges and quotas defined by the VAX/VMS operating system.

4.4.2 Input, Output, and Error Streams

At login the system also establishes equivalences for default process logical names, including SYS$INPUT, SYS$OUTPUT, and SYS$ERROR. These logical names provide permanent associations for the process's input, output, and error streams. For an interactive process, these logical names are initially equated to the terminal and are used by the command interpreter to read command lines and to display output and error messages.

These logical names are also available to all images that the process executes. For example, a program that performs explicit input/output requests through RMS (Record Management Services) macros or system services can write records to SYS$OUTPUT. If you execute this image interactively, the output is directed to your current terminal; if you execute the image in a batch job, the output is directed to the batch job output log.

4.4.3 Processes and Subprocesses

A process can execute only one program image at a time. Some applications may require concurrent execution of cooperating programs. Because the system uniquely identifies every process an image executing in one process can communicate with or control another process by referring to that process's identification number.

Processes executing with the same group number in their UICs can also refer to one another by process name, that is, a character string name assigned to the process. When you first log into the system, the system gives your process the same name as your user name. If you log in at more than one terminal, processes after the first are given names based on the name of the terminal at which you logged in.

Most processes in the system are detached processes; that is, they execute independently of one another. The process that the system creates for you at login is a detached process. An image executing in a process can create another type of process, called a subprocess. The process that creates a subprocess is called the owner process.

The owner of a subprocess can:

- Define the base priority, privileges, and resource quotas that the subprocess will have.
- Specify the name of an executable image to be executed in the subprocess, and control the execution of the image.
- Obtain information about the status of the subprocess and the system resources it has used.

Generally, a subprocess executes a single image, and when the image exits, the system deletes the subprocess. If the owner process is deleted (for example, if you log out) and a subprocess still exists, the system also deletes the subprocess.

If you plan to develop an application to use cooperating processes, you should be familiar with the VAX/VMS system services that provide process communication and control functions. These services are described in detail in the VAX/VMS System Services Reference Manual.

In conjunction with the system services that control processes and subprocesses, you can use the RUN command to create a subprocess to execute a particular image. For details, see the description of the RUN (Process) command in Part II.

CHAPTER 5

GRAMMAR RULES

This chapter describes the syntax rules for the VAX/VMS command language, including rules for:

- Entering commands
- Entering file specifications
- Entering qualifiers
- Entering character string data
- Entering numeric values
- Forming expressions
- Specifying lexical functions
- Entering date and time values

5.1 RULES FOR ENTERING COMMANDS

A command string is the complete specification of a command, including the command name, command qualifiers, parameters, and file qualifiers if any.

The general format of a command is:

```
[$] [label:]  command name[/qualifiers...] parameter[/qualifiers...][...]
```

Because you can continue a command on more than one line, the term command string is used to define the entire command that is passed to the system. You can precede any command string with a dollar sign ($) character. In interactive mode the command interpreter ignores the dollar sign. However, it is required in batch mode and in command procedures (even those executed interactively).

Each item in a command must be properly delimited, as follows:

- At least one blank character must separate the command name from the first parameter, and at least one blank must separate each additional parameter from the previous parameter. Multiple blanks and tabs are permitted in all cases where a single blank is required.

- Each qualifier must be preceded with a slash (/). The slash can be preceded by or followed by any number of blanks or tabs.

- If a label precedes the command, the label must be terminated with a colon (:). The colon can be preceded by or followed by any number of blanks or tabs.

In addition, any special characters you enter on a command can be treated as delimiters, depending on the context of the command. These characters are listed in Table 5-1.

5.1.1 **Rules for Continuing Commands on More than One Line**

The maximum number of characters accepted as a command string from one line is 132. However, you can enter a command string on more than one line by using the continuation character, a hyphen (-), as the last element on a command line.

Command line continuation is especially useful when you enter a command and want to specify many qualifiers, or when you place a command in a command procedure file and want to make the procedure more readable. For example:

```
$ PRINT MYFILE -
 /AFTER=17:00 -
 /COPIES=20 -
 /NAME="COMGUIDE"
```

Note that when you continue a command, you must still provide the required space before each parameter.

There is no restriction on the number of continued lines you can use to enter a command. However, the maximum number of characters that you can enter in a command string depends on the expansion of all lexical functions and symbols in the command line. The string that results must not exceed 255 characters. Furthermore, there is a limit on the complexity of commands that include a large number of lexical functions and symbols. You can avoid problems with the complexity limitation if you construct complex commands from a series of concatenated symbols.

NOTE

Some DCL commands invoke RSX-11M utility programs. These commands are actually "back translated" to form command strings for the RSX-11M utility. The maximum length of these lines, after they have been back translated, is 80 characters.

5.1.2 **Rules for Entering Comments**

Indicate a comment by preceding it with an exclamation mark (!). Comments are valid in the following positions:

- As the first item on a command line; in this case, the entire line is considered a comment and is not processed except for symbol substitutions and lexical functions.

- Following the last character in a command string, or after a hyphen in a command line

Some examples of valid placement of comments follow:

```
$ !THIS ENTIRE LINE IS A COMMENT
$ PRINT MYFILE - !  PRINT COMMAND COMMENT
$_/COPIES=3     !  3 COPIES, PLEASE
```

When you enter a command interactively and continue the command on more than one line, the command interpreter uses the $_ prompt to indicate that it is still accepting the command.

5.1.3 Rules for Truncating Keywords

All keywords that you enter as command input can be abbreviated by truncating characters on the right. You can truncate:

- Command names
- Command keyword parameters
- Qualifiers
- Qualifier keyword values

When the command interpreter reads a command line, it only examines the first four characters of each keyword you type.

5.1.3.1 **Truncating Command Names** - All command name keywords are unique when truncated to their first four characters. You may truncate command names to fewer characters as long as you ensure the truncation is unique. For example, the TYPE command is currently the only command that begins with the character "T". Therefore, the TYPE command could be truncated to just one character. The DEALLOCATE and DEASSIGN commands, however, have the same first three characters, so these commands cannot be truncated to fewer than four characters.

Exceptions: The following commands are exceptions to the minimum truncation rule because they can be truncated to just their first character, even though other commands begin with the same character:

```
CONTINUE
DEPOSIT
EXAMINE
RUN
```

5.1.3.2 **Truncating Command Parameters, Command Qualifiers, and Command Qualifier Values** - All keywords recognized by individual commands (that is, command parameters and qualifiers and qualifier values) are unique with respect to the other keywords recognized by the same command. This means that you can also abbreviate these elements to four or fewer characters. For example, the option TRACEBACK on the /DEBUG qualifier in some commands can be abbreviated to TRAC (or even T) with no conflict.

Note that some qualifiers permit a negative form. For example, /NOLIST is the negative form of the /LIST qualifier. In applying the minimum four-character truncation rule, do not count the NO prefix as the first two of the four characters. In this case, the minimum truncation that guarantees uniqueness is /NOLIST. The slash character

(/) is also disregarded in counting characters. However, the underscore character (_) is counted, as in /[NO]D_LINES, where the minimum truncation for guaranteed uniqueness is /D_LI and /NOD_LI.

5.1.3.3 Abbreviations in Command Procedure Files

- Special considerations apply when you type commands in command procedure files. It is recommended that you use the full names of all the above components to ensure readability. If you do abbreviate any of these items, you should never abbreviate to fewer than the four-character truncation described above, or you risk the possibility that your command procedure may not be compatible with future releases of the system.

5.2 RULES FOR ENTERING FILE SPECIFICATIONS

The format of VAX/VMS file specifications is described in detail in Chapter 2, "File Specifications and Logical Names." Use this format for all files you specify as parameters to DCL commands. Remember that all file specifications that you enter as parameters to system commands or as qualifier values can be subject to logical name translation and the application of defaults.

Some commands perform only a single level of logical name translation; that is, translation is not recursive. When this is the case, the parameter description indicates that fact.

In many cases, a command applies a unique default file type to input or output file specifications. The parameter descriptions indicate the default file type, if any.

The parameter descriptions also indicate whether you can specify wild card characters in a field in the file specification.

5.2.1 Rules for Entering File Specification Lists

An input file parameter for many commands has the format:

```
     file-spec[,...]
```

This format indicates that you can enter more than one file specification. You can separate the file specifications with commas (,) or plus signs (+). However, commas and plus signs sometimes have different meanings as separators. The parameter description always states how the command interprets the list: in most cases, commas and plus signs are equivalent.

Any number of blanks or tab characters can precede or follow the commas or plus signs. The list of file specifications is treated as a single parameter; the system applies temporary defaults to the files specified in the list.

5.3 RULES FOR ENTERING QUALIFIERS

Commands can take one or both of the following types of qualifier:

- Command qualifiers
- File qualifiers

Command qualifiers have the same meaning regardless of whether they appear following the command name or following a command parameter. For example:

```
$PRINT/HOLD SPRING.SUM,FALL.SUM
$PRINT SPRING.SUM,FALL.SUM/HOLD
```

The /HOLD qualifier is a command qualifier; therefore, the two PRINT commands shown above are equivalent. Both files are placed in a hold status.

File qualifiers, however, sometimes have different meanings depending on where you place them in the command. If specified immediately after a file specification parameter, they affect only the file they follow. If specified after the command name, they affect all files specified as parameters.

For example:

```
$PRINT/COPIES=2  SPRING.SUM,FALL.SUM
$PRINT  SPRING.SUM/COPIES=2,FALL.SUM
```

The first PRINT command shown above requests two copies of each of the files SPRING.SUM and FALL.SUM. The second PRINT command requests two copies of the file SPRING.SUM, but only one copy of FALL.SUM.

Some file qualifiers can be applied only to the specification of a parameter and cannot be specified following the command name. When this is the case, the qualifier description indicates that the qualifier is associated with a file parameter. In all other cases, if you specify a file qualifier following the command name, the qualifier is applied to all files specified (if you specify more than one).

5.3.1 Rules for Determining Qualifier Defaults

Qualifiers fall into the general categories described below. The values shown on the right are defaults.

- Qualifiers with simple positive and negative forms. For example:

  ```
  /DELETE
  /NODELETE
  ```

  These qualifiers are listed in the command format boxes in Part II, with their default value, as follows:

  ```
  /[NO]DELETE                /NODELETE
  ```

- Qualifiers that require a numeric or character string variable or keyword value. For example:

  ```
  /COPIES=n
  ```

  The default values for these qualifiers are shown in Part II as:

  ```
  /COPIES=n                    /COPIES=1
  ```

- Qualifiers that accept a file specification value and that also have a negative form. For example:

  ```
  /DEBUG
  /NODEBUG
  /DEBUG[=file-spec]
  ```

  These qualifiers are listed in the command format boxes in Part II as follows:

  ```
  /[NO]DEBUG[=file-spec]       /NODEBUG
  ```

  Defaults are noted, where they exist and can be readily described. In this case, default always means the action taken when you omit the entire qualifier, not just the optional part of it. Look in the corresponding text for a description of what happens if you specify a qualifier without an optional portion. If the default action is too complicated to express in simple terms in the format section, you must see the qualifier text for a full description.

  NOTE

  Certain qualifiers that request output files accept file specification values and apply defaults for these specifications in a special way, as described in Section 5.3.3.

- Qualifiers that are overridden by other qualifiers. For example:

  ```
  /PROCESS
  /GROUP
  /SYSTEM
  ```

  The default action is shown in Part II with each qualifier in the list, as follows:

Qualifier	**Default**
/GROUP	/PROCESS
/PROCESS	/PROCESS
/SYSTEM	/PROCESS

- Qualifiers that affect command execution only if explicitly present and have no corresponding default. For example:

  ```
  /RSX11
  ```

  No defaults are given for this type of qualifier. It is a required qualifier.

If you specify the same qualifier more than once when you enter a command, or specify both a positive and negative form of the same qualifier or qualifiers that override one another, the command interpreter accepts only the last specification. For example:

```
$ PRINT MYFILE /COPIES=3/BURST/COPIES=2/NOBURST
```

For this PRINT command, the command accepts only the /COPIES=2 and the /NOBURST qualifiers.

If you enter conflicting qualifiers for a command, the command interpreter generally issues an error message. If a qualifier conflicts with another qualifier, the descriptions of these qualifiers indicate the conflict.

5.3.2 **Rules for Entering Qualifier Values**

Many qualifiers accept keywords, file specifications, character strings, or numeric values. For keywords, follow the rules for truncating keywords (Section 5.1.3); for other types of value, follow the rules for entering file specifications (Section 5.2), character data (Section 5.4), or numeric values (Section 5.5).

You must separate a qualifier and value with either an equal sign (=) or colon (:). For example, the following specifications are equivalent:

```
/OUTPUT=DBA1:NEW.DAT       /OUTPUT:DBA1:NEW.DAT
```

Many qualifiers accept one or more keyword or variable values. The syntax is represented in the format as:

```
/qualifier=value
```

For example:

```
/COPIES=3
/OVERRIDE=EXPIRATION
```

When more than one value is accepted, the syntax is:

```
/qualifier=value[,...]
```

If you want to specify more than one value for a qualifier, you must separate the values with commas and enclose them in parentheses. For example:

```
/PARAMETERS=(3,"CYGNUS,LYRA",PRINT)
```

The second parameter is enclosed in quotation marks because it contains an embedded comma.

Some qualifier keyword values require additional data. Separate the keyword from its data with a colon or an equal sign. For example:

```
/PROTECTION=GROUP:RW
/PROTECTION:GROUP:RW
/PROTECTION=GROUP=RW
/PROTECTION:GROUP=RW
```

These expressions are all equivalent. To specify multiple keywords that require values, enclose the list in parentheses. For example:

```
/BLOCKS=(START:0,END:10)
/PROTECTION=(GROUP:RW,OWNER:RWD)
```

5.3.3 Rules for Entering Output File Qualifiers

Some qualifiers request output from a command and optionally accept a file specification value. For example, the /LIST and /OBJECT qualifiers for the compilers, as well as the /EXECUTABLE qualifier for the linker, are output file qualifiers that fit into this category.

The default file specification for output files requested by these qualifiers depends on the placement of the qualifier in the command. The rules are:

1. If the qualifier is present by default, the output file specification defaults to the current default device and directory, and the name of the first input file. The qualifier provides a default file type. Some examples are:

Command	Output File
LINK A	A.EXE
LINK A,B	A.EXE
LINK [TEST]A,[]B	A.EXE
LINK A.OBJ	A.EXE

2. If the qualifier is present following the command name, and the qualifier does not specify an output file specification, the output file specification defaults to the current default device and directory, and the name of the first input file. The qualifier provides a default file type. Some examples are:

Command	Output File
LINK/EXE A	A.EXE
LINK/EXE A,B	A.EXE
LINK/EXE A.OBJ	A.EXE

3. If the qualifier is present following the specification of an input file, and the qualifier does not specify an output file specification, the output file specification defaults to the device, directory, and file name of the immediately preceding input file. The qualifier provides a default file type. Some examples are:

Command	Output File
FORTRAN A,B/LIST	B.LIS
FORTRAN A+C/LIST,B/LIST+D	C.LIS and B.LIS
FORTRAN [MAL]A/LIST+[]B	[MAL]A.LIS
LINK A+[TEST]D/EXE	[TEST]D.EXE

4. If the qualifier specifies a file specification for the output file, then any fields entered in the file specification are used to name the output file, and no default file name is supplied. Some examples are:

Command	**Output File**
LINK A+B/EXE=C	C.EXE
FORTRAN/LIST=A B+C	A.LIS
FORTRAN/LIST=A B,C	A.LIS (2 versions)
LINK/EXE=[TEST] A	[TEST].EXE

In all cases, the version number of the output file is always one greater than any existing file with the same file name and file type.

Note that when a logical name is used for any input file specifications, the entire equivalence name of the logical name is used to determine the output file specification. If a logical name is used for an input file and its equivalence name contains a file type, then the same file type is used for the output file.

5.4 RULES FOR ENTERING CHARACTER STRING DATA

When you enter commands, you can use any combination of uppercase and lowercase letters. The command interpreter translates lowercase letters to uppercase letters and compresses multiple blank spaces or tabs to a single blank, except when a character string is enclosed in quotation marks (for example, "This is a string.").

Enclose character string data in quotation marks when the string contains any of the following and you do not want the command interpreter to translate them:

- Literal lowercase letters
- Required multiple blanks or tab characters
- Any nonalphanumeric character that has special significance to the command interpreter

The alphanumeric characters are:

A through Z

a through z

0 through 9

$ (dollar sign)

_ (underscore)

Table 5-1 lists the nonalphanumeric characters the command interpreter recognizes and describes their meanings. The characters described as "reserved special characters" can be used in character strings without being enclosed in quotation marks. Other characters may require quotation marks depending on the context of the command. When in doubt, use quotation marks.

Table 5-1
Nonalphanumeric Characters

Symbol	Name	Meaning
@	At sign	Places the contents of a command procedure file in the command input stream
:	Colon	Device name delimiter in a file specification; a double colon (::) is a node name delimiter Qualifier value delimiter; separates a qualifier name from its value Symbol name delimiter in a character string assignment Label delimiter Delimiter between shared memory name and object name
/	Slash	Qualifier delimiter Division operator in an arithmetic expression
+	Plus sign	Parameter concatenation operator A unary plus sign or addition operator in an arithmetic expression
,	Comma	List element separator for parameters or for qualifier value lists
-	Hyphen	Continuation character A unary minus sign or subtraction operator in an arithmetic expression A directory searching wild character
( )	Parentheses	List delimiters for qualifier value list Operation precedence indicators in an arithmetic expression Lexical function argument delimiters
[ ]	Square brackets	Directory name delimiters in a file specification Substring specification delimiter in an assignment statement

(continued on next page)

Table 5-1 (Cont.)
Nonalphanumeric Characters

Symbol	Name	Meaning
< >	Angle brackets	Directory name delimiters in a file specification
?	Question mark	Reserved special character
&	Ampersand	Execution-time substitution operator; otherwise, a reserved special character
\	Back slash	Reserved special character
=	Equal sign	Qualifier value delimiter; separates a qualifier name from its value Arithmetic or string assignment operator
^	Circumflex	Reserved special character
#	Number sign	Reserved special character
*	Asterisk	Wild card character in a file specification Multiplication operator in an arithmetic expression Abbreviation delimiter in a symbol definition
'	Apostrophe	Substitution operator
.	Period	File type and version number delimiter in file specifications Logical or comparison operator delimiter in an expression A subdirectory delimiter
;	Semicolon	Version number delimiter in a file specification
%	Percent sign	Radix operator Wild card character in a file specification
!	Exclamation point	Comment delimiter
"	Quotation mark	Literal string delimiter

5.5 RULES FOR ENTERING NUMERIC VALUES

The command interpreter treats all literal numeric values as decimal integers. To specify numeric values in commands, you can use one of the following radix operators preceding a numeric value to indicate the radix (number base):

Operator	Meaning
%D	Decimal
%X	Hexadecimal
%O	Octal

For example:

```
%D19 = %X13 = %O23
%X1F = %D31 = %O37
%O20 = %X10 = %D16
```

There cannot be any blanks between a radix operator and a value.

5.6 RULES FOR FORMING EXPRESSIONS

When the command interpreter evaluates an expression, it assigns a value based on the result of the operations specified in the expression. If the expression contains logical operators or arithmetic or string comparison operators, the expression is considered true if it results in an odd numeric value; the expression is considered false if it results in an even numeric value.

The following sections show how to use expressions in assignment statements. Symbols defined by assignment statements can be tested in IF commands. The rules defined below also apply to the use of expressions in the IF command.

5.6.1 **Rules for Entering Operators**

Logical and comparison operators must be preceded by a period (.) with no intervening blanks. The operator must be terminated with a period. You can type any number of blanks or tabs between operators and operands. For example, the following expressions are equivalent:

```
A.EQS.B
A .EQS.  B
```

Each operator (with the exception of .NOT.) must have operands on each side.

5.6.2 **Rules for Specifying Operands**

The command interpreter performs symbol substitution on all operands in expressions that are not enclosed in quotation marks.

Table 5-2 summarizes the valid operators you can use in expressions and gives the precedence of each operator. Operators of precedence 6 are performed first, while operators of precedence 1 are performed last. When you form expressions, you can use parentheses to define the order of precedence in evaluation.

The valid data types you can specify for operands for each category of operator are summarized below:

Operands for logical operations can be:

- Literal numeric values
- Symbol names
- Expressions

Operands for arithmetic comparisons can be:

- Literal numeric values
- Symbol names
- Literal character strings that begin with any of the letters Y, N, T, or F, or symbol names equated to character strings beginning with any of those letters

Operands for string comparisons can be:

- Literal character strings
- Symbol names equated to literal character strings
- Literal numeric values

Operands for arithmetic operations can be:

- Literal numeric values
- Symbol names equated to numeric values
- Expressions that result in numeric values

Expressions are particularly useful in the context of command procedures developed for specific applications. For details on how to create and use command procedures and for examples of using expressions and symbolic values, see the VAX/VMS Guide to Using Command Procedures.

Table 5-2
Summary of Operators in Expressions

Operator		Precedence	Operation
Logical Operators	.OR.	1	Logical OR
	.AND.	2	Logical AND
	.NOT.	3	Logical complement
Arithmetic Comparison Operators	.EQ.	4	Arithmetic equal to
	.GE.	4	Arithmetic greater than or equal to
	.GT.	4	Arithmetic greater than
	.LE.	4	Arithmetic less than or equal to
	.LT.	4	Arithmetic less than
	.NE.	4	Arithmetic not equal to

(continued on next page)

Table 5-2 (Cont.)
Summary of Operators in Expressions

Operator		Precedence	Operation
String Comparison Operators	.EQS.	4	String equal to
	.GES.	4	String greater than or equal to
	.GTS.	4	String greater than
	.LES.	4	String less than or equal to
	.LTS.	4	String less than
	.NES.	4	String not equal to
Arithmetic Operators	+	5	Arithmetic sum
	-	5	Arithmetic difference
	*	6	Arithmetic product
	/	6	Arithmetic quotient

5.7 RULES FOR SPECIFYING LEXICAL FUNCTIONS

You can use lexical functions in any context in which you normally use symbols or expressions. The general format of a lexical function is:

```
'F$function-name([args,...])
```

'F$
: indicates that what follows is a lexical function. All three characters, including the substitution operator ('), are required.

function-name
: specifies the function to be evaluated. All function names are keywords. You can truncate function names to any unique truncation, remembering that keeping four characters will guarantee uniqueness in all future releases of VMS.

( )
: 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. You can specify arguments using symbol names, numeric literals, or string literals enclosed in quotation marks. The command interpreter assumes that all strings beginning with alphabetic letters that are not enclosed in quotation marks are symbol names. If a symbol is undefined, the command interpreter substitutes a null string.

  Function arguments cannot specify string substitution or other lexical functions.

For a complete list of the lexical functions, their formats, and the arguments required by each, see the entry in PART II entitled "Lexical Functions." For additional details on how to use the lexical functions, see the VAX/VMS Guide to Using Command Procedures.

5.8 RULES FOR ENTERING DATES AND TIMES

When a command accepts a qualifier that specifies a time value, the time value is either an absolute time or a delta time:

- An absolute time is a specific date and time of day, for example 10-JUN-1978 10:53:22.10.
- A delta time is a future offset from the current date and time of day, for example 2 days and 3 hours from now.

The syntax rules for specifying time values are described below.

5.8.1 **Absolute Times**

Absolute times generally have the format:

```
[dd-mmm-yyyy[:]][hh:mm:ss.c]
```

You can specify either the date or the time, or both. The variable fields are as follows:

Field	**Meaning**
dd	Day of month (1 through 31)
mmm	Month; the month must be specified as one of the following three-character abbreviations: JAN, FEB, MAR, APR, MAY, JUN, JUL, AUG, SEP, OCT, NOV, DEC
yyyy	Year
hh	Hour of the day (0 through 23)
mm	Minute of the hour (0 through 59)
ss	Seconds (0 through 59)
c	Hundredths of seconds (0 through 99)

You may also specify one of the following keywords wherever an absolute time is appropriate:

Keyword	**Meaning**
TODAY	The current day, month, and year at 00:00:00.0 o'clock
TOMORROW	24 hours after 00:00:00.0 o'clock today
YESTERDAY	24 hours before 00:00:00.0 o'clock today

5.8.1.1 **Syntax** - The punctuation marks in an absolute time indicate how the system interprets the time value you enter, as follows:

1. If you specify both the date (dd-mmm-yyyy) and the time (hh:mm:ss.c), you must type the colon between the date and the time.

2. You can truncate either the date or the time on the right; however, if you are specifying both a date and a time, the date part must contain at least one hyphen.

3. You can omit any of the fields within the date or time, as long as you type the punctuation marks; the system supplies default values.

4. The period between seconds and hundredths of seconds is a delimiter; it is not a decimal point.

5.8.1.2 Defaults

5.8.1.2 **Defaults** - When the date or any of its fields is omitted from an absolute time value, the system supplies the current day, month, or year by default.

When any field is omitted from the time, the system supplies a value of 0 for the field.

5.8.1.3 **Examples** - Some examples of specifying absolute times are:

Time Specification	Result
28-JUN-1978:12	12:00 noon on June 28, 1978
28-JUN	Midnight (00:00 o'clock) at the beginning of the 28th of June, this year
15	3:00 p.m., today
15-	The 15th day of the current month and year, at midnight
18:30	6:30 p.m., today
15--::30	00:30 o'clock, on the 15th day of the current month
00:00:00.2	Two-hundredths of a second after midnight, today

Note that whenever you issue a command and specify an absolute time that has already passed, the system executes the specified action immediately.

5.8.2 Delta Times

Delta times have the format:

```
[dddd-][hh:mm:ss.c]
```

The variable fields are as follows:

Field	Meaning
dddd	Number of days, 24-hour units (0 through 9999)
hh	Number of hours (0 through 23)
mm	Number of minutes (0 through 59)
ss	Number of seconds (0 through 59)
c	Number of hundredths of seconds (0 through 99)

5.8.2.1 **Syntax** - When you specify a delta time value, you can truncate the time field on the right. You can also omit any of the variable fields, as long as you supply the punctuation marks.

5.8.2.2 **Defaults** - When any field is omitted from a delta time value, the system supplies a value of 0 for the field.

5.8.2.3 **Examples** - Some examples of specifying delta times are:

Time Specification	Result
3-	3 days from now (72 hours)
3	3 hours from now
:30	30 minutes from now
3-:30	3 days and 30 minutes from now
15:30	15 hours and 30 minutes from now

5.9 RULES FOR SYMBOL SUBSTITUTION

This section explains the symbol substitution process, including what happens when certain types of symbols have no value.

This topic is expanded with numerous examples in the <u>VAX/VMS Guide to Using Command Procedures</u>.

5.9.1 Phases of Symbol Substitution

The command interpreter substitutes symbols during three distinct phases of command processing. The three phases occur in the following sequence:

1. Lexical Processing. The command interpreter scans the command and locates the symbol at the beginning of a command string, except where the next nonblank character is an equal sign (=) or a colon (:). It also locates symbol names that are delimited with apostrophes ('), including those occurring

within comments. All substitutions are performed according to the sequence in Section 5.9.2. Note that any of these symbols that remain undefined after this step are treated as null strings.

2. Parsing. Symbol names preceded with ampersands (&) are substituted during command parsing. If one of these symbols is undefined, it is treated as a null string within the context of the command that will be executed.

3. Execution. Symbol names may occur in contexts where substitution is automatic at execution time. Examples include: symbols used with IF, DEPOSIT, EXAMINE, or WRITE commands, symbols appearing on the right-hand side of arithmetic assignment statements, or symbols in lexical functions. If any of these symbols cannot be defined, an error occurs.

5.9.2 Order of Symbol Substitution

Each time a substitution occurs, the substitutions are made from left to right on each line in the command.

In performing symbol substitution, the command interpreter searches the local symbol table for the current command level for a match and replaces the symbol name with the numeric value or character string assigned to it. If the symbol is not found, the command interpreter searches the local symbol tables of preceding command levels, in reverse order. Last, it searches the global symbol table.

5.10 RULES FOR PROTECTION CODES

Section 3.1.1 defines the user categories and access types employed in protection codes. This section provides the rules for specifying protection codes.

Any combination of the access types, READ, WRITE, EXECUTE, and DELETE, can be specified for any of the four categories of user, SYSTEM, OWNER, GROUP, or WORLD. The following syntax rules apply:

- When you specify a protection code, you must abbreviate access types to one character. User categories can be entered in full or truncated to any number of characters.
- You can specify the user categories and access types in any order.
- If you omit an access type for a user category, that category of user is denied that type of access.
- When you specify a protection code, separate each user category from access type with a colon.
- If you specify more than one user category, separate the categories with commas and enclose the entire code in parentheses.
- When you omit a user category from a protection code applied to an entire volume, that category of user is denied all types of access.

- When you omit a user category from a protection code applied to one or more files or from a code specified for the default protection, the current access allowed that category of user remains unchanged.
- When you specify a user category with a protection code of 0 or with a null protection code, you deny that user category any access. (Be certain to omit the colon after the user category, if you want to specify a null protection code.)

For example:

```
$ SET PROTECTION=(SYSTEM:RWED,GROUP:R,WORLD)/DEFAULT
```

This protection code allows the system all types of access, group members read access only, prohibits all access by users in the world category, and does not change the current access for the owner.

5.11 RULES FOR DEFINING AND ABBREVIATING COMMAND SYMBOLS

You can use abbreviated forms of symbols for commands if you define them using the abbreviation punctuation character, the asterisk (*). For example, to abbreviate a local symbolic name, DISPLAY, define it as:

```
$ DISP*LAY  := $DISPLAY
```

Then, the DISPLAY utility will be executed whenever the following versions of the symbolic name are used:

```
DISP
DISPL
DISPLA
DISPLAY
```

Generally, you can use abbreviated symbol definitions in any situation that allows any symbol to be used. However, there are some restrictions:

- You cannot abbreviate symbols unless you are defining local or global character strings.
- You cannot abbreviate symbols that involve substring replacement
- When you define abbreviated symbols, existing symbols may be deleted, or become ambiguous. If an existing symbol exactly matches the new symbol at or past the abbreviation punctuation (the asterisk), the existing symbol is deleted from the symbol table. If an existing symbol matches the new symbol up to the asterisk, the new symbol is not entered in the symbol table.

PART II

COMMAND DESCRIPTIONS

= (Assignment Statement)

Defines a symbolic name for a character string or for an arithmetic value or expression.

Formats

```
symbol-name =[=] expression

symbol-name :=[=] string

symbol-name[bit-position,size] =[=] expression[1]

symbol-name[offset,size] :=[=]  string[1]
```

Command Qualifiers	Defaults
None.	None.

Prompts

None.

Command Parameters

symbol-name

Defines a 1- through 255-alphanumeric character-string name for the symbol. The symbol name must begin with an alphabetic character (uppercase and lowercase characters are equivalent).

If you specify a single equal sign (=) in the assignment statement, the symbol name is placed in the local symbol table for the current command level.

If you specify double equal signs (==) in the assignment statement, the symbol name is placed in the global symbol table. Global symbols are recognized in any command procedure as well as at the interactive command level. There cannot be any blanks between the equals signs or between an equals sign and a colon.

expression

Specifies a numeric value or an arithmetic or logical expression to be equated to the symbol name. If you specify an expression, the command interpreter analyzes the expression, substituting symbol values if necessary, before making the assignment.

1. In this syntax, the bold brackets in the expressions [bit-position,size] and [offset,size] are required. The brackets in the expressions =[=] and :=[=] indicate that the second equal sign is optional.

For a summary of operators and details on the syntax requirements and how to specify expressions, see Section 5.6, "Rules for Forming Expressions."

string

Specifies a character-string value (or any expression resulting in a character string value) to be equated to the symbol. A character string can have from 0 through 255 characters. The string can contain any alphanumeric or special characters; enclose it in quotation marks if it contains any multiple blanks or tab characters, lowercase letters, an exclamation point (!), or quotation marks ("). To specify a string that contains literal quotation marks, enclose the entire string in quotation marks, and use a double set of quotation marks within the string. For example:

```
$ HELLO := "PATTI SAYS ""HI"""
```

You can specify a null string either by using a double set of quotation marks with no intervening characters or by specifying no string. For example:

```
$ NULLSTRING := ""
```

You can omit a trailing quotation mark on the end of a line.

If a string beginning with a dollar sign is not enclosed in quotation marks, the command interpreter assumes that the string is the file specification of an executable image. When the symbol-name thus defined appears as the first item in a command string, the command interpreter executes the image. The file specification must include a device name.

[bit-position,size]

Defines a range within the current value of symbol-name that is to be stored with the binary value of the indicated expression. You can specify any position within a numeric value as the starting bit position. If you specify a size value that exceeds 32, the value 32 is used.

[offset,size]

Defines the substring location at which the current value of symbol-name is to be overlaid with characters from the indicated string expression.

offset

Specifies the position relative to the beginning of the symbol-name's string value at which replacement is to begin. If the offset is greater than the length of the string, the resulting string is filled with blanks to the requested offset before replacement occurs. The maximum value you can specify is 255.

size

Specifies the number of characters, beginning with the first character, in the string expression to extract. If the size is greater than the length of the string, the string is blank-filled on the right to the requested size before it is used to overlay the symbol-name string. The value of the size plus the offset must not exceed 255.

You can specify the offset and size using literal numeric values or arithmetic expressions, including expressions consisting of lexical functions.

When you specify a substring expression in an assignment statement, there cannot be any blanks preceding or following the substring expression.

Description

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

Creating a Symbol that Executes an Image: When a symbol is equated to a string that begins with a dollar sign ($) followed by a file specification, the command interpreter assumes that the file specification is that of an executable image. It further assumes the file has a device and directory of SYS$SYSTEM and a default file type of EXE. Example 5 that follows illustrates this technique for creating command synonyms. Appendix A elaborates further on this topic and its application. Section 5.11 includes a description of how to designate acceptable abbreviations for your command synonyms.

Reserved Symbols: The following symbol names are reserved. They are global symbols under the control of the operating system and cannot be explicitly redefined:

$STATUS	Return status from the most recently executed command or program, or status value specified by a command procedure when it exited.
$SEVERITY	Severity level of the status code from the most recently executed command or program. The possible values are:

Value	**Severity Level**
0	Warning
1	Success
2	Error
3	Informational
4	Severe, or fatal error

You can use these symbolic names to test the completion status of command or program execution to determine the success or failure of a request, and optionally to provide for conditional processing based on the value returned.

Symbol Substitution: Rules for symbol substitution appear in Section 5.9. For further discussion of symbol substitution and examples of how to use symbols in command procedures, see the <u>VAX/VMS Guide to Using Command Procedures</u>.

Examples

1.
```
$ TIME := SHOW TIME
$ TIME
  28-JUL-1978 11:55:44
```

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.
```
$ LIST :== DIRECTORY
$ TIME :== SHOW TIME
$ QP :== SHOW QUEUE/DEVICE
$ SS :== SHOW SYMBOL
```

The file, SYNONYM.COM, contains the assignment statements shown; these are user-defined synonyms for commands. Execute this command procedure as follows:

```
@SYNONYM
```

The global symbol definitions are made, and you can now use these synonyms at the interactive command level. Note that the assignments are global; otherwise, the symbol names would be deleted after the file SYNONYM.COM completed execution.

3.
```
$ COUNT = 0
$ LOOP:  COUNT = COUNT + 1
    .
    .
    .
$ IF COUNT.LT.5 THEN GOTO LOOP
```

The symbol COUNT is initially assigned a numeric value of 0; a loop is established to increment the value of COUNT by 1 each time the loop is entered. Note that when the symbol name COUNT appears on the right-hand side of an arithmetic assignment statement, the command interpreter automatically substitutes its current value.

The IF command tests the value of COUNT; if less than 5, the procedure branches to the label LOOP and the statements between the label LOOP through the IF command are executed again. When the value of the symbol count reaches 5, the loop is not executed again and the command following the IF command is executed.

4.

```
$ NAME := MYFILE
$ TYPE := .DAT
$ PRINT 'NAME''TYPE'
```

In this example the symbol NAME is equated to a file name and the symbol TYPE is equated to a file type. The PRINT command refers to both of these symbol names to form a file specification. The apostrophes surrounding each symbol name indicate that these are symbols to be substituted.

The PRINT command prints the file MYFILE.DAT.

5.

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

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, the file has a file type of EXE. Thus, the symbol STAT in this example becomes a synonym for the command:

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

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

6.

```
$ WRITE SYS$OUTPUT "Beginning Test No.  ''COUNT'"
```

The WRITE command writes a line of data to the current output stream. The string to be written is enclosed in quotation marks so that the lowercase letters will not be translated to uppercase. However, the string contains the name of a symbol, COUNT, which must be substituted with its current value before the line is written.

The symbol COUNT is preceded with two apostrophes and terminated with a single apostrophe to request that symbol substitution be performed.

7.

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

This command procedure uses a counter to refer to parameters that are passed to it. Up to eight parameters, named P1, P2, and so on, can be passed. 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 symbol COUNT is substituted with its current value and 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. The substitution of P1, P2, and so on, is automatic within the context of the IF command, because the IF command tests symbolic and literal expressions.

The APPEND and DELETE commands, however, 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. When these commands are initially scanned each time through the loop, COUNT is substituted with its current value. Then, when the commands execute, the & causes another substitution: P1, P2, and so on. You can invoke this procedure with the line:

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

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

8.

```
$ FILE_SPEC := 'P1'
$ LOC = 'F$LOCATE(".",FILE_SPEC)
$ FILE_NAME := 'F$EXTRACT(0,LOC,FILE_SPEC)
```

These lines show how to extract the file name portion of a string containing a file name, file type, and optionally a file version number.

The first statement equates the symbol FILE_SPEC to the parameter P1, which must be passed to the command procedure. Note that the apostrophes are required; otherwise, the symbol name FILE_SPEC is equated to the literal string P1.

The second statement uses the lexical function F$LOCATE to locate the period within the file specification string. The function returns, in the symbol LOC, a numeric value representing the offset of the period within the string value of FILE_SPEC.

The third statement uses the symbol name LOC to specify how many characters of the file specification are to be extracted. The lexical function F$EXTRACT requests that LOC characters, beginning at the beginning of the string, be extracted from the current value of the string named FILE_SPEC. The result is equated to the symbol FILE_NAME.

If this procedure is passed the parameter MYFILE.DAT, the resulting value of the symbol LOC is 6 and the resulting value of the symbol FILE_NAME is the string MYFILE.

9.

```
$ FILE_NAME[0,2]:= OL
```

The substring expression in the assignment statement overlays the first two characters of a file name string 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.

If the current value of the symbol FILE_NAME is MYFILE when this assignment statement is executed, the resulting value of the symbol name is OLFILE.

10.
```
$ FILE_TYPE := .TST
$ FILE_NAME['F$LENGTH(FILE_NAME),4)]:= 'FILE_TYPE'
```

In this example, the symbol name FILE_TYPE is equated to the string .TST. The second assignment statement uses the lexical function F$LENGTH to define the offset value in the substring expression.

The F$LENGTH lexical function returns the length of the string equated to the symbol FILE_NAME; thus the substring expression requests that 4 characters of the string currently equated to the symbol FILE_TYPE be placed at the end of the string currently equated to FILE_NAME. If the current value of FILE_NAME is MYFILE, the F$LENGTH lexical function returns a value of 6 and the substring expression is:

```
$ FILE_NAME[6,4]:= 'FILE_TYPE'
```

Thus, the resultant value of the string 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]]]
```

Command Qualifiers	Defaults
/OUTPUT=file-spec	None.

Prompts

None.

Command Parameters

file-spec

Specifies the command procedure to be executed, or the device or file from which input for the preceding command is to be read.

If you do not specify a file type, the system uses the default file type of COM.

No wild card characters are allowed in the file specification.

p1 [p2 ... p8]

Specify from one to eight optional parameters to pass to the command procedure. The parameters assign numeric or character string values to the symbols named P1, P2, and so on in the order of entry, to a maximum of 8. The symbols are local to the specified command procedure. The command interpreter sets all unspecified parameters to null strings.

Separate each parameter with one or more blanks. You can specify a numeric or character string value using any alphanumeric or special characters, with the following restrictions:

1. If the first parameter begins with a slash character (/), you must enclose the parameter in quotation marks (").

2. To pass a parameter that contains embedded blanks or literal lowercase letters, place the parameter in quotation marks.

3. To pass a parameter that contains literal quotation marks, enclose the entire string in quotation marks and use a double set of quotation marks within the string. For example:

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

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

   ```
   Never say "quit"
   ```

Description

Use command procedures to catalog frequently used sequences of commands. A command procedure can contain:

- Any valid DCL command. All commands must begin with a dollar sign ($) character. If a command is continued with the continuation character (-), the subsequent lines must not begin with a $.

- Data. Any line in a command procedure that does not contain a dollar sign in the first character position (or 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.

- Qualifiers and/or parameters for a specific command. If the file contains only parameters for the command, the @ command must be preceded by a space. If the file contains qualifiers for the command, the @ command must not be preceded with a space. If the file contains only parameters and/or qualifiers, the lines must not begin with dollar signs ($). Any additional data on the command line following the @file-spec is treated as parameters for the procedure.

A command procedure can also contain a request to execute another command procedure. The maximum level to which command procedures can be nested is eight.

Command procedures can also be queued for processing as batch jobs, either with the SUBMIT command or by placing a deck of cards containing the command procedure in the system card reader. Batch jobs submitted through the card reader must be preceded with JOB and PASSWORD commands.

For more information and examples of creating and submitting command procedures for execution, see the VAX/VMS Guide to Using Command Procedures.

Command Qualifiers

/OUTPUT=file-spec

Requests that all output directed to the logical device SYS$OUTPUT be written to the file or device specified. System responses and error messages are written to SYS$COMMAND as well as to the specified file.

If you specify /OUTPUT, the qualifier must immediately follow the file specification of the command procedure. Otherwise, it is interpreted as a parameter to pass to the command procedure.

The default output file type is LIS.

No wild card characters are allowed in the file specification.

Examples

1.

```
$ ON WARNING THEN EXIT
$ IF P1.EQS."" THEN INQUIRE P1 FILE
$ FORTRAN/LIST 'P1'
$ LINK 'P1'
$ RUN 'P1'
$ PRINT 'P1'
```

This command procedure, named DOFOR.COM, executes the FORTRAN, LINK, and RUN commands to compile, link, and execute a program whose file specification is passed as a parameter. The ON command requests that the procedure not continue if any of the commands results in warnings or errors. The IF command checks to see if a parameter was passed; if not, the INQUIRE command issues a prompting message to the terminal and equates what you enter with the parameter P1. You can execute this procedure as follows to compile, link, run, and obtain a listing of the program AVERAGE.FOR:

```
$ @DOFOR AVERAGE
```

2.

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

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

3.

```
$ RUN 'P1' -
  /BUFFER_LIMIT=1024 -
  /FILE_LIMIT=4 -
  /PAGE_FILES=256 -
  /QUEUE_LIMIT=2 -
  /SUBPROCESS_LIMIT=2 -
  'P2'  'P3'  'P4'  'P5'  'P6'  'P7'  'P8'
```

This procedure, named SUBPROCES.COM, issues the RUN command to create a subprocess. It contains qualifiers defining quotas for subprocess creation. For example, the procedure can be invoked as follows:

```
$ @SUBPROCES  LIBRA  /PROCESS_NAME=LIBRA
```

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

4.
```
$ ASSIGN SYS$COMMAND:  SYS$INPUT:
$ NEXT:
$ INQUIRE NAME "File name"
$ IF NAME.EQS."" THEN EXIT
$ EDIT/SOS 'NAME' .DOC
$ GOTO NEXT
```

This procedure, named EDOC.COM, invokes the SOS 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 of SYS$INPUT for the procedure. This allows the editor to read input from the current command device (the terminal) rather than from the input stream (the command procedure file).

ALLOCATE

Provides exclusive access to a device and optionally establishes a logical name for the device. Once a device has been allocated, other users cannot access the device until you specifically deallocate it or log out.

Format

```
ALLOCATE  device-name[:]  [logical-name[:]]
```

Command Qualifiers	Defaults
None.	None.

Prompts

Device: device-name

Log_name: logical-name

Command Parameters

device-name

Specifies the name of the device to be allocated. The device name can be a generic device name, such that if no controller or unit number is specified, the system allocates any available device that satisfies those components of the device name that are specified.

logical-name

Specifies a 1- through 63-character logical name to be associated with the device. The logical name is placed in the process logical name table, with the name of the physical device allocated as its equivalence name. Subsequent references to the logical name result in automatic translation to the specified device name.

If you specify a trailing colon (:) on the logical name, the colon is removed from the name before the name is placed in the logical name table.

Examples

1.
```
$ ALLOCATE  _DMB2:
  _DMB2: ALLOCATED
```

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

2.
```
$ ALLOCATE  MT:   TAPE:
  _MTB2: ALLOCATED
    .
    .
    .

$ SHOW LOGICAL TAPE
  TAPE = _MTB2:     (process)
$ DEALLOCATE TAPE
$ DEASSIGN TAPE
```

The ALLOCATE command requests the allocation of any tape device whose name begins with MT, to be assigned the logical name, TAPE. The ALLOCATE command locates an available tape device and responds with the name of the device allocated. 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.

ANALYZE

Provides a description of the contents of an object file or the symbol information appended to a shareable image file. In describing the records, ANALYZE also identifies certain errors.

Format

```
ANALYZE  file-spec

Command Qualifiers                     Defaults

/IMAGE                                 /OBJECT
/[NO]INTERACTIVE                       /NOINTERACTIVE
/OBJECT                                /OBJECT
/OUTPUT[=file-spec]

File Qualifiers                        Defaults

/DBG                                   None.
/EOM
/GSD
/MHD
/TBT
/TIR
```

Prompts

File: file-spec

Command Parameters

file-spec

Specifies the name of the object or shareable image file you want analyzed. By default, the file type is assumed to be OBJ, unless you specify the /IMAGE qualifier. The /IMAGE qualifier imposes a default file type of EXE.

No wild card characters are allowed in the file specification.

Description

ANALYZE provides a description of the records comprising an object or shareable image file. It also performs a partial error analysis on the file. This command is intended primarily for use by programmers of compilers, debuggers, or other software that involves VAX/VMS object modules. For a full description of the use of the ANALYZE Utility in conjunction with the Linker, see the VAX-11 Linker Reference Manual.

By default, if you specify none of the file qualifiers (/DBG, /EOM, /GSD, and so forth), you obtain the same results as if you had specified all of them. However, as soon as you specify one of them, you disable the others and then must explicitly request all those file qualifiers you want in effect.

Command Qualifiers

/IMAGE

Specifies that the analysis should occur on the symbol information appended to a shareable image file. This qualifier and the /OBJECT qualifier are mutually exclusive. If you omit this qualifier, the default is /OBJECT.

/INTERACTIVE
/NOINTERACTIVE

Controls whether the analysis occurs interactively with the user at a terminal. The interactive mode provides a display of the analysis of each record in sequence and gives the user the opportunity to respond Y (Yes) or N (No) to the question of whether to continue the interactive analysis after each record is displayed.

The default is /NOINTERACTIVE.

/OBJECT

Specifies that the analysis should occur on an object file. This is the default. Furthermore, the /OBJECT and /IMAGE qualifiers are mutually exclusive.

/OUTPUT[=file-spec]

Identifies the output file for storing the results of the analysis. By default, this file receives a file type of ANL. If you omit the file name, the output is directed to a file with the same name as the input file and a file type of ANL.

No wild card characters are allowed in the file specification.

File Qualifiers

/DBG

Specifies that the analysis should include all debugger information records.

/EOM

Specifies that the analysis should include all end-of-module records.

/GSD

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

/MHD

Specifies that the analysis should include all module header records.

/TBT

Specifies that the analysis should include all traceback records.

/TIR

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

Examples

1. `$ ANALYZE/IMAGE/OUTPUT=SYS$OUTPUT TAXES.EXE`

   Analyzes all the records in the shareable image file TAXES.EXE (including, by default, the debugger information, end-of-module, global symbol, module header, traceback, and text information and relocation records). Notice that this logical name for the output file causes the output to appear on the user's terminal. However, the display occurs in total, so there is no halting after each record display to request permission to continue, as there is in the interactive mode.

2. `$ ANALYZE/OBJECT/INTERACTIVE TAXES/TBT`

   This command interactively analyzes all traceback records in the object file TAXES.OBJ.

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

Format

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

Command Qualifiers          Defaults

/[NO]LOG                    /NOLOG

File Qualifiers             Defaults

/ALLOCATION=n
/[NO]CONTIGUOUS             /NOCONTIGUOUS
/EXTENSION=n
/FILE_MAXIMUM=n
/[NO]NEW                    /NONEW
/PROTECTION=code
/[NO]READ_CHECK             /NOREAD_CHECK
/[NO]WRITE_CHECK            /NOWRITE_CHECK
```

Prompts

From: input-file-spec[,...]

To: output-file-spec

Command Parameters

input-file-spec[,...]

Specifies the names of one or more input files to be appended.

If you specify more than one input file, separate the specifications with either commas (,) or plus signs (+). Commas and plus signs are equivalent; all files specified are appended, in the order specified, to the end of the output file.

You can use full wild carding in the file specification as described in Section 2.1.6.

output-file-spec

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

You must specify at least one field in the output file specification. If you do not specify a device and/or directory, the APPEND command uses your current default device and directory. For other fields that you do not specify, the APPEND command uses the corresponding field of the input file specification.

If you specify the asterisk (*) wild card character in any field(s) of the output file specification, the APPEND command uses the corresponding field of the related input file specification.

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. You can use the /NEW qualifier to request that if the output file does not exist, the APPEND command should create it.

Command Qualifiers

/LOG
/NOLOG

Controls whether the APPEND command displays the file specifications of each file appended.

If you specify /LOG, the APPEND command displays, after each append operation, the file specifications of the input and output files, and the number of blocks or the number of records appended. At the end of command processing, the APPEND command displays the number of new files created.

File Qualifiers

/ALLOCATION=n

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

This qualifier is valid only if /NEW is specified, and the allocation size is applied only if a new file is actually created. If a new file is created and you do not specify /ALLOCATION, the initial allocation of the output file is determined by the size of the input file.

/CONTIGUOUS
/NOCONTIGUOUS

Indicates whether the output file is contiguous, that is, whether the file must occupy consecutive physical disk blocks.

By default, the APPEND command creates an output file in the same format as the corresponding input file. 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 files are contiguous.

/EXTENSION=n

Specifies the number of blocks to be added to the output file each time the file is extended.

The extension value is applied only if a new file is actually created. If you specify /EXTENSION, the /NEW qualifier is assumed.

/FILE_MAXIMUM=n

Specifies the maximum number of logical records that the output file can contain.

This qualifier is valid only for new relative files. If you specify /FILE_MAXIMUM, the /NEW qualifier is assumed.

/NEW
/NONEW

Controls whether the APPEND command creates a new file. By default, the output file specified must already exist. Use /NEW to request that if the specified output file does not already exist, the APPEND command should create it.

/PROTECTION=code

Defines the protection to be applied to the output file.

Specify the protection code using the standard rules given in Section 5.10. Any protection attributes not specified are taken from the current protection of the output file; or, if a new file is created, from the current default protection.

/READ_CHECK
/NOREAD_CHECK

Requests the APPEND command to read each record in the input file(s) twice to verify that all records were correctly read.

/WRITE_CHECK
/NOWRITE_CHECK

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

Examples

1.
```
$ APPEND   TEST.DAT   NEWTEST.DAT
```

   The APPEND command appends the contents of the file TEST.DAT from the default disk and directory to the file NEWTEST.DAT.

2.
```
$ APPEND/NEW/LOG *.TXT   MEM.SUM
%APPEND-I-CREATED, DBA2:[MAL]MEM.SUM;1 created
%APPEND-S-COPIED, DBA2:[MAL]A.TXT;2 copied to DBA2:[MAL]MEM.SUM;1 (1 block)
%APPEND-S-APPENDED, DBA2:[MAL]B.TXT;3 appended to DBA2:[MAL]MEM.SUM;1 (3 records)
%APPEND-S-APPENDED, DBA2:[MAL]G.TXT;7 appended to DBA2:[MAL]MEM.SUM;1 (51 records)
%APPEND-S-NEWFILES, 1 file created
```

   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, DBA2:[MAL]A.DAT;4 appended to DBA2:[MAL]C.DAT;4 (2 records)
%APPEND-S-APPENDED, DBA2:[MAL]B.MEM;5 appended to DBA2:[MAL]C.DAT;4 (29 records)
```

   The input file specifications in this example request APPEND to append separate files. The APPEND command appends the files A.DAT and B.MEM to the file C.DAT.

4.
```
$ APPEND/LOG A.*   B.*
%APPEND-S-APPENDED, DBA2:[ANK]A.DAT;5 appended to DBA2:[ANK]B.DAT;1 (5 records)
%APPEND-S-APPENDED, DBA2:[ANK]A.DOC;2 appended to DBA2:[ANK]B.DAT;1 (1 record)
```

   Both the input and output file specifications contain wild card characters in the file type field. The APPEND command appends each file with a file name of A to a file with a file name of B; the file type of the first input file located determines the output file type.

Equates a logical name to a physical device name; to a complete file specification, or to another logical name; and places the equivalence name string in the process, group, or system logical name table.

Format

```
ASSIGN   equivalence-name[:]  logical-name[:]


Command Qualifiers         Defaults

/GROUP                     /PROCESS
/PROCESS                   /PROCESS
/SUPERVISOR_MODE           /SUPERVISOR_MODE
/SYSTEM                    /PROCESS
/USER_MODE                 /SUPERVISOR_MODE
```

Prompts

Device: equivalence-name

Log_name: logical-name

Command Parameters

equivalence-name

Specifies the name of the device or file specification to be assigned a logical name.

If you specify a physical device name, terminate the device name with a colon (:).

You can specify a logical name for any portion of a file specification. If the logical name translates to a device name, and will be used in place of a device name in a file specification, terminate it with a colon (:).

logical-name

Specifies a 1- through 63-character logical name to be associated with the device. If you terminate the logical name with a colon, the system 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 logical name contains any characters other than alphanumeric characters or delimiters not recognized within device names, enclose it in quotation marks.

If the logical name already exists in the specified logical name table, the new definition supersedes the old definition, and the system displays an informational message indicating that fact.

Description

If you enter more than one of the qualifiers /PROCESS, /GROUP, or /SYSTEM, or both of the qualifiers /SUPERVISOR_MODE and /USER_MODE, only the last one entered is accepted.

For additional information on how to create and use logical names, see Section 2.2, "Logical Names."

Command Qualifiers

/GROUP

Places the logical name and its associated device name in the group logical name table. Other users with the same group number in their UICs (user identification codes) can access the logical name.

The user privilege GRPNAM is required to place a name in the group logical name table.

/PROCESS

Places the logical name and its associated device name in the process logical name table. This is the default.

/SUPERVISOR_MODE

Specifies, for an entry in the process logical name table, that the logical name be entered in supervisor mode.

This is the default for the process logical name table entries. The /SUPERVISOR_MODE qualifier is ignored when entries are made in the group or system logical name tables.

/SYSTEM

Places the logical name and its associated device name in the system logical name table. Any user can access the logical name.

The user privilege SYSNAM is required to place a name in the system logical name table.

/USER_MODE

Specifies, for an entry in the process logical name table, that the logical name be entered in the user mode.

A user mode logical name is typically used for the execution of a single image. For example, it allows an image executing in a command procedure to read a different SYS$INPUT than that in use by the command procedure. User mode entries are deleted when any image executing in the process exits (that is, after any DCL command or user program that executes an image completes execution), or when a STOP command is issued.

By default, process logical name table entries are made in supervisor mode. The /USER_MODE qualifier is ignored when entries are made in the group or system logical name tables.

Examples

1.
```
$ ASSIGN DBA2:[CHARLES]  CHARLIE
$ PRINT  CHARLIE:TEST.DAT
```

The ASSIGN command associates the logical name CHARLIE with the directory name CHARLES on the disk DBA2. Subsequent references to the logical name CHARLIE result in the correspondence between the logical name CHARLIE and the disk and directory specified. Thus, the PRINT command queues a copy of the file DBA2:[CHARLES]TEST.DAT to the system printer.

2.
```
$ ASSIGN DBA1:  TEMP:
$ SHOW LOGICAL TEMP
  TEMP  =  DBA1:     (process)
$ DEASSIGN TEMP
```

The ASSIGN command equates the logical name TEMP to the device DBA1. 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 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.

3.
```
$ MOUNT  MTB3:  MASTER  TAPE
$ ASSIGN  TAPE:NAMES.DAT  PAYROLL
$ RUN  PAY
    .
    .
    .
```

The MOUNT command establishes the logical name TAPE for the device MTB3, 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, a subsequent OPEN request in a program that refers 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.

4.
```
$ ASSIGN/GROUP  _DBB1:  GROUP_DISK:
```

The ASSIGN command assigns the logical name GROUP_DISK to the physical device DBB1. Subsequently, another user in the same group can issue the command:

```
    $ ASSIGN  GROUP_DISK:[HIGGINS]WEEKLY.OUT  OUTFILE
```

This ASSIGN command equates the logical name OUTFILE to a file on the device specified by the logical name GROUP_DISK. When the ASSIGN command executes, it locates the logical name GROUP_DISK in the group logical name table and translates it to the device name DBB1.

5.
```
$ ASSIGN/PROCESS/GROUP  DBA1:  SYSFILES:
$ SHOW LOGICAL  SYSFILES
  SYSFILES  =  DBA1:     (group)
```

The ASSIGN command contains conflicting qualifiers. The response from the SHOW LOGICAL command indicates that the name was placed in the group logical name table.

6.
```
$ ASSIGN/GROUP 'F$LOGICAL("SYS$COMMAND") TERMINAL
  PREVIOUS LOGICAL NAME ASSIGNMENT REPLACED
```

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

7.
```
$ ASSIGN/USER_MODE SYS$COMMAND:  SYS$INPUT:
$ EDIT AVERAGE.FOR
$ FORTRAN AVERAGE
$ LINK AVERAGE
$ RUN AVERAGE
55
55
9999
```

In the command procedure illustrated above, the ASSIGN command equates the logical name SYS$INPUT with SYS$COMMAND for the execution of the next command. When the SOS editor is invoked, it will read all input from the current terminal, regardless of the number of active command levels. When the edit session is terminated, the deassignment is automatically made, and the procedure goes on to compile, link, and run the program AVERAGE.

The program AVERAGE reads input from the current default input device: in this example, test data records follow the RUN command in the input stream. Note that, if the assignment of SYS$INPUT were not made in user mode, it would not be as convenient to re-establish the default relationship between the current input stream and the logical device named SYS$INPUT.

Invokes the VAX-11 BASIC[1] compiler to compile a BASIC program.

Format

```
BASIC  [file-spec[,...]]
```

Command Qualifiers	Defaults
None.	None.

File Qualifiers	Defaults
/[NO]CHECK[=(option[,...])]	/CHECK=(BOUNDS,OVERFLOW)
/[NO]CROSS_REFERENCE	/NOCROSS_REFERENCE
/[NO]DEBUG[=(option[,...])]	/DEBUG=TRACEBACK
/DOUBLE	/SINGLE
/[NO]LINE	/LINE
/[NO]LIST[=file-spec]	
/LONG	/LONG
/[NO]MACHINE_CODE	/NOMACHINE_CODE
/[NO]OBJECT[=file-spec]	
/SCALE=n	/SCALE=0
/SINGLE	/SINGLE
/WORD	/LONG

Prompts

None.

Command Parameters

file-spec[,...]

Specifies one or more VAX-11 BASIC source programs to be compiled. If the file specification does not contain a file type, the compiler uses the default file type of BAS.

You can specify more than one input file. If you separate the file specifications with commas (,), each file is compiled separately. If you separate the file specifications with plus signs (+), the files are appended (see the description of APPEND in the VAX-11 BASIC User's Guide) and compiled as a single input file, producing single object and listing files. If you specify SYS$INPUT as the file-spec parameter, the source program must follow the command in the input stream. In this case, both the object module file (given by the /OBJECT qualifier) and the listing file (given by the /LIST qualifier) must be explicitly named.

1. Available under separate license.

No wild card characters are allowed in the file specification.

If you do not provide a source file specification, VAX-11 BASIC responds with the prompt Ready and expects you to enter a VAX-11 BASIC source program interactively. See the VAX-11 BASIC User's Guide for more details. Note, however, that without a file specification, none of the file qualifiers is permitted.

File Qualifiers

/CHECK[=(option[,...])]
/NOCHECK

Controls whether the compiler produces extra code to check on program correctness at run time. You can request the following options:

ALL	Provides both BOUNDS and OVERFLOW checks.
[NO]BOUNDS	Produces code to check that all array references are to addresses within the array boundaries.
NONE	Provides no checking.
[NO]OVERFLOW	Enables integer overflow traps to detect arithmetic overflow on fixed point calculations involving integer data types.

By default, if you omit either the /CHECK qualifier or specify /CHECK without options, both integer overflow and bounds checking will occur (equivalent to /CHECK=ALL). Note that /NOCHECK is equivalent to /CHECK=NONE.

If you specify more than one option, separate them with commas and enclose the list in parentheses.

/CROSS_REFERENCE
/NOCROSS_REFERENCE

Controls whether a cross reference listing is included in the listing file. The /CROSS_REFERENCE qualifier includes a cross reference listing and therefore requires that a listing file exist. The default is /NOCROSS_REFERENCE, which excludes it.

/DEBUG[=option[,...])]
/NODEBUG

Controls whether the compiler makes local symbol table and traceback information available to the debugger and the run time error reporting mechanism.

You can request the following options:

ALL	Provides both local symbol table and traceback information.
NONE	Does not provide either local symbol table or traceback information.
[NO]SYMBOLS	Provides the debugger with local symbol definitions for user-defined variables.

[NO]TRACEBACK	Provides the debugger with compiler-generated line numbers so that the debugger and error reporting mechanisms can translate virtual addresses into source program subroutine names and line numbers.

By default, if you completely omit the /DEBUG qualifier, the compiler produces only traceback information (equivalent to /DEBUG=(NOSYMBOLS,TRACEBACK). However, if you specify /DEBUG without any options, the default is both SYMBOLS AND TRACEBACK (equivalent to /DEBUG=ALL).

Note that /NODEBUG is equivalent to /DEBUG=NONE.

If you specify more than one option, separate them with commas and enclose the list in parentheses.

For details on how to debug a VAX-11 BASIC program with the VAX-11 Symbolic Debugger, see the VAX-11 BASIC User's Guide.

/DOUBLE

Causes all floating point numbers to be 64 bits wide. The qualifiers /DOUBLE and /SINGLE are mutually exclusive. If you omit /DOUBLE, all floating point numbers will be 32 bits wide, by default.

/LINE
/NOLINE

Enables the error processor to determine the line numbers of statements with errors and to output these line numbers for the user. By default, line numbers will be provided with error notifications. For more details on this feature, see the VAX-11 BASIC User's Guide.

/LIST[=file-spec]
/NOLIST

Controls whether the compiler creates a listing file.

If you issue the BASIC command from interactive mode, the compiler, by default, does not create a listing file. If the BASIC command is executed from batch mode, /LIST is the default. If you omit the file specification, the VAX-11 BASIC compiler gives the listing file the same file name as the input source file and a file type of LIS.

When you specify /LIST, you can control the defaults applied to the output file specification by the placement of the qualifier in the command, as described in Section 5.3.3, "Rules for Entering Output File Qualifiers."

No wild card characters are allowed in the file specification.

/LONG

Causes all untyped integers to be 32 bits long. Note that the /LONG and /WORD qualifiers are mutually exclusive. The default is /LONG.

/MACHINE_CODE
/NOMACHINE_CODE

Controls whether the listing produced by the compiler includes the machine language code generated by the compiler.

By default, the compiler does not include machine language code in the listing. The /MACHINE_CODE qualifier is ignored if /LIST is not specified, either explicitly or by default.

/OBJECT[=file-spec]
/NOOBJECT

Controls whether the compiler creates an output object module.

By default, the compiler produces an object module with the same file name as the source file and a file type of OBJ. When you specify /OBJECT, you can control the defaults applied to the output file specification by the placement of the qualifier in the command, as described in Section 5.3.3, "Rules for Entering Output File Qualifiers."

No wild card characters are allowed in the file specification.

/SCALE=n

Specifies a scale factor for double precision numbers to afford compatibility with BASIC PLUS. The scale factor n can be in the range of 0 through 6. By default, the scale factor is 0, that is, there is no scaling.

/SINGLE

Causes all floating point numbers to be 32 bits wide, which is also the default. The /SINGLE and /DOUBLE qualifiers are mutually exclusive.

/WORD

Causes all untyped integers to be 16 bits long. The /LONG and /WORD qualifiers are mutually exclusive. By default, untyped integers will be 32 bits long, so you must specify /WORD if you want untyped integers to be 16 bits long.

Examples

1.
```
$ BASIC/LIST/OBJECT/MACHINE_CODE CALCAGE
```

Compiles the VAX-11 BASIC program CALCAGE.BAS, producing a listing and an output object module. The listing file is named CALCAGE.LIS and includes the machine code output. The object module is named CALCAGE.OBJ. By default, integer overflow and bounds checking will occur, line numbers will appear with error messages, traceback will be enabled, and floating point numbers and untyped integers will both be 32 bits long.

2.
```
$ BASIC

VAX-11 BASIC    V1-001

Ready
```

The BASIC command without a file specification indicates a terminal session is desired. VAX-11 BASIC responds with a Ready message.

BASIC/RSX11

Invokes the PDP-11 BASIC-PLUS-2/VAX[1] compiler to begin a BASIC session. All subsequent command input is read by BASIC-PLUS-2. The /RSX11 qualifier is required.

Format

BASIC/RSX11

Additional Command Qualifiers	Defaults
None.	None.

Prompts

Basic2

Command Parameters

None.

Description

After invoking PDP-11 BASIC-PLUS-2/VAX, use BASIC subcommands to create, edit, and compile BASIC programs.

To link and run a BASIC program, you must issue the BUILD subcommand to request BASIC to create a command procedure suitable for input to the RSX-11M Task Builder. After exiting from BASIC with the EXIT subcommand, create the executable image file by specifying the command procedure as input to the task builder.

For details on how to use BASIC-PLUS-2, see the BASIC-PLUS-2 RSX-11M/IAS User's Guide.

1. Available under separate license.

Examples

1.
```
$ BASIC/RSX11

Basic Plus 2 V01-60

Basic2

OLD AVERAG

Basic2

COMPILE

Basic2

BUILD/SEQUENTIAL

Basic2

EXIT

$ MCR TKB @AVERAG

$ RUN AVERAG
```

The BASIC/RSX11 command invokes BASIC-PLUS-2. The OLD, COMPILE, and BUILD subcommands define the input file, AVERAG.B2S, and request BASIC to compile the file and to create a command procedure for input to the task builder. The BUILD command creates the file AVERAG.CMD.

The MCR TKB command invokes the task builder to create an executable image file, using the commands in the file AVERAG.CMD. The RUN command executes the image AVERAG.EXE created by the task builder.

2.
```
$ CREATE CPYFIL.B2S
  .
  .
  .

$ BASIC/RSX11
OLD CPYFIL
COMPILE
BUILD/SEQUENTIAL
EXIT
$ MCR TKB @CPYFIL
$ ASSIGN TEST.DAT INFILE
$ ASSIGN TEST.OUT OUTFILE
$ RUN CPYFIL
$ TYPE TEST.OUT
```

This command procedure uses the CREATE command to create the BASIC source file, assigning it the name of CPYFIL.B2S. When the BASIC/RSX11 command executes, it reads subsequent input from the command input stream. After BASIC executes the OLD, COMPILE, and BUILD subcommands, the EXIT command terminates the BASIC session; the next commands are read by the DCL command interpreter.

After the MCR TKB command creates the image file, the ASSIGN commands assign equivalence names to the logical file names INFILE and OUTFILE. (These files must be referred to in BASIC OPEN statements in the file CPYFIL.EXE.) The RUN command executes the image CPYFIL.EXE and the TYPE command verifies the program output in the file TEST.OUT.

The command procedure can be created interactively and submitted for execution as a batch job with the SUBMIT command, or punched on cards and submitted to a system card reader preceded with cards containing JOB and PASSWORD commands.

Invokes the VAX-11 BLISS-32[1] compiler to compile one or more BLISS-32 or common BLISS source programs. This command is described in detail in the VAX-11 BLISS-32 User's Guide.

Format

```
BLISS  file-spec[,...]

Command Qualifiers                    Defaults

None.                                 None.

File Qualifiers                       Defaults

/[NO]CODE                             /CODE
/[NO]DEBUG                            /NODEBUG
/[NO]LIBRARY[=file-spec]              /NOLIBRARY
/[NO]LIST[=file-spec]                 (see text)
/MACHINE_CODE_LIST[=(option[,...])]   /MACHINE_CODE_LIST=(NOASSEMBLER,-
                                         BINARY,COMMENTARY,-
                                         OBJECT,SYMBOLIC,-
                                         NOUNIQUE_NAMES)
/[NO]OBJECT[=file-spec]               /OBJECT
/OPTIMIZE[=(option[,...])]            /OPTIMIZE=(LEVEL:2,NOQUICK,-
                                         SAFE,SPACE)
/[NO]QUICK                            /NOQUICK
/SOURCE_LIST[=(option[,...])]         /SOURCE_LIST=(NOEXPAND_MACROS,-
                                         HEADER,NOLIBRARY,-
                                         PAGE_SIZE:52,-
                                         NOREQUIRE,SOURCE,-
                                         NOTRACE_MACROS)
/TERMINAL[=(option[,...])]            /TERMINAL=(ERRORS,NOSTATISTICS)
/[NO]TRACEBACK                        /TRACEBACK
/VARIANT[=n]                          /VARIANT=0
```

Prompts

File: file-spec[,...]

Command Parameters

file-spec[,...]

Specifies one or more VAX-11 BLISS-32 or common BLISS source program files to be compiled. If you do not specify a file type for an input file, BLISS-32 uses the default file type of B32 as a first choice and if that fails, uses BLI as the second choice.

1. Available under separate license.

You can specify more than one input file. If you separate the file specifications with commas (,), each file is compiled separately. If you separate the file specifications with plus signs (+), the files are concatenated and compiled as a single input file, producing single object and listing files. If you specify SYS$INPUT as the file-spec parameter, the source program must follow the command in the input stream. In this case, both the object module file (given by the /OBJECT qualifier) and the listing file (given by the /LIST qualifier) must be explicitly named.

No wild card characters are allowed in the file specification.

File Qualifiers

/CODE
/NOCODE

Specifies whether or not the compiler should produce executable code.

Use /NOCODE to perform syntax checking of a source program; because the compiler does not produce code, the compilation speed is increased.

By default, executable code is produced.

/DEBUG
/NODEBUG

Indicates whether or not the compiler should produce a symbol table that may be used with the debugger.

By default, no debug symbol table is produced.

/LIBRARY[=file-spec]
/NOLIBRARY

Produces a library file rather than an object file.

By default, no library is produced. The default file type for the library file is L32.

No wild card characters are allowed in the file specification.

/LIST[=file-spec]
/NOLIST

Controls whether an output listing is created and optionally provides an output file specification for the listing file.

When in batch mode, the output listing is created by default. However, in interactive mode the default is to produce no output listing.

The default file type for listing files is LIS.

No wild card characters are allowed in the file specification.

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

Directs the compiler how to format the object part of the output listing.

You can request the following options:

[NO]ASSEMBLER	Indicates whether or not the assembler instructions produced as a result of the compilation will be listed. ASSEMBLER directs that the assembler instructions be given and that all other information be included within comments. NOASSEMBLER is the default value; it suppresses the assembler instructions.
[NO]BINARY	Indicates whether or not to include a listing of the binary for each instruction in the object code listing. The default is BINARY, which includes the binary code in the listing.
[NO]COMMENTARY	Indicates whether or not to include a commentary field in the object code listing. The default is COMMENTARY, which produces machine-generated commentary (limited to a cross-reference).
[NO]OBJECT	Indicates whether or not the object part of the listing should be produced. OBJECT produces the object portion of the listing, and is the default value. NOOBJECT suppresses this portion of the listing.
[NO]SYMBOLIC	Directs whether or not to include a machine code listing that uses names from the BLISS source program. NOSYMBOLIC omits the names, while SYMBOLIC, the default, includes them.
[NO]UNIQUE_NAMES	Directs whether or not the compiler should produce unique names for OWN variables and nonglobal ROUTINE names when it creates a listing that is to be assembled. NOUNIQUE_NAMES is the default, and it suppresses the production of unique names.

/OBJECT[=file-spec]
/NOOBJECT

Controls whether an object module is created by the BLISS compiler, and optionally provides an output file specification for the file.

By default, the compiler creates an object module with the same file name as the first input file and a file type of OBJ. When you specify /OBJECT, you can control the defaults applied to the output file specification by the placement of the qualifier in the command, as described in Section 5.3.3, "Rules for Entering Output File Qualifiers."

No wild card characters are allowed in the file specification.

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

Defines the degree and type of optimization so that the compiler can select the appropriate optimization strategies.

You can request the following options:

LEVEL:n	Controls the optimization level. The four possible values for n are: 0, 1, 2, or 3. To request minimum optimization, specify a level of 0. Maximum optimization occurs for the option LEVEL:3. By default, the optimization level is 2.
[NO]QUICK	Increases the compilation speed by omitting some standard optimizations. The default is NOQUICK, which does not affect compilation speed.
[NO]SAFE	Indicates whether or not named variables in the source code will be addressed only by name. SAFE, the default, dictates that variables will be addressed only by name. Use NOSAFE to indicate that variables are addressed by pointers, and not only by name.
SPEED SPACE	SPEED optimizes for program execution speed by using more storage. The default is SPACE, which is the opposite of SPEED. With the SPACE option, storage space is conserved, which may entail sacrifices in execution speed.

If you specify more than one option, separate them by commas and enclose the list in parentheses.

/QUICK
/NOQUICK

Controls whether the compilation speed is increased by omitting some standard optimizations. The default is /NOQUICK, which does not affect compilation speed. The /QUICK qualifier has the same effect as the QUICK option of the /OPTIMIZE qualifier.

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

Defines one or more source value options for the compilation.

You can request the following options:

[NO]EXPAND_MACROS	Specifies whether or not to include the expansion of each macro call in the listing file. EXPAND_MACROS includes the expansions in the listing file. NOEXPAND_MACROS is the default; it omits the macro expansions from the listing file.

[NO]HEADER	Specifies whether or not the source listing will be paged and include headings. HEADER pages the source program listing and provides a heading on each page. HEADER is the default choice. NOHEADER omits the headings, does not page the listing, and omits the statistics in the compilation summary.
[NO]LIBRARY	Specifies whether or not to produce a trace identifying the libraries and their contributions. LIBRARY produces a trace in the source listing file that identifies the library after a LIBRARY declaration and the first use of each name whose definition is obtained from a library file. NOLIBRARY is the default value; it does not produce a trace of the libraries and their contributions.
PAGE_SIZE:n	Specifies the number of lines allowed for each page in the listing file. You must choose a number greater than 19. The default value of PAGE_SIZE is 52 lines.
[NO]REQUIRE	Determines whether or not to include the contents of all require files in the listing. REQUIRE requests that the contents of the require files be included. NOREQUIRE is the default choice; it omits the contents of the require files from the listing.
[NO]SOURCE	Determines whether to increment or decrement the listing control counter. Output is listed when the listing control counter is positive and not listed when the counter is zero or negative. SOURCE, the default value, increments the counter. NOSOURCE decrements the counter.
[NO]TRACE_MACROS	Determines whether the listing should include a trace of macro expansions. TRACE_MACROS includes the trace of the expansion of each macro call in the listing file. It includes the parameter binding and any intermediate forms of expansion, as well as the result of the expansion. NOTRACE_MACROS omits the trace of the macro expansions and is the default value.

If you specify more than one option, separate them by commas and enclose the list in parentheses.

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

Controls the output sent to the terminal device during a compilation. Error messages and certain statistics are optional information that can be directed to the terminal.

You can request the following options:

[NO]ERRORS	Determines whether or not to list each error message on the terminal as encountered in the compilation. ERRORS is the default; it lists the errors on the terminal. NOERRORS omits the error messages.
[NO]STATISTICS	Determines whether or not statistics should appear on the terminal during compilation. STATISTICS lists the name and size of each routine on the terminal after each routine is compiled. NOSTATISTICS is the default; it omits the routine names and sizes.

If you specify more than one option, separate them by commas and enclose the list in parentheses.

/TRACEBACK
/NOTRACEBACK

Controls whether the compiler generates information in the object module that can be used by the debugger to locate module, routine, and PSECT names.

The default is NOTRACEBACK, which produces the minimum size object module. No information is provided for debugging or tracing.

/VARIANT[=n]

Specifies the value of the predeclared literal %VARIANT.

If no value is specified for n, the default value of 1 is used; otherwise, %VARIANT assumes the specified value.

If /VARIANT is not specified, %VARIANT has the value of 0.

For additional details on these functions, see the VAX-11 BLISS-32 User's Guide.

Examples

1.
```
$ BLISS/LIST=WEATHER2    WEATHER
```

   Compiles the source program WEATHER.B32, producing an object module named WEATHER.OBJ and a listing file named WEATHER2.LIS. Note that all the default options are selected, so that the compiler performs normal optimization, balances the time and space trade-off in favor of space, and addresses all variables by name. %VARIANT assumes a value of 0.

CANCEL

Cancels scheduled wakeup requests for a specified process. This includes wakeups scheduled with the RUN command and with the Schedule Wakeup ($SCHDWK) system service.

Format

```
CANCEL [process-name]

Command Qualifiers                Defaults

/IDENTIFICATION=process-id        None.
```

Prompts

None.

Command Parameters

process-name

Specifies the 1- to 15-alphanumeric character string name of the process for which wakeup requests are to be canceled. Process names are assigned to processes when they are created. The specified process must have the same group number in its user identification code (UIC) as the current process.

If you also specify the /IDENTIFICATION qualifier, the process name is ignored. If you specify neither the process-name parameter nor the /IDENTIFICATION qualifier, the CANCEL command cancels scheduled wakeup requests for the current (that is, the issuing) process.

Description

The user privilege GROUP is required to cancel scheduled wakeups for non-owned processes in the same group; the user privilege WORLD is required to cancel scheduled wakeups for any process in the system.

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

Command Qualifiers

/IDENTIFICATION=process-id

Specifies the process identification number the system assigned to the process when the process was created. When you specify the process identification, you can omit leading zeroes.

Examples

1.
```
$ RUN/SCHEDULE=14:00 STATUS
%RUN-S-PROC_ID, identification of created process is 0013012A
   .
   .
   .

$ CANCEL/IDENTIFICATION=13012A
```

The RUN command 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 wakeup request.

2.
```
$ 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 creates a subprocess named LIBRA to execute the image LIBRA.EXE at hourly intervals.

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

CLOSE

Closes a file that was opened for input or output with the OPEN command and deassigns the logical name specified when the file was opened.

Format

```
CLOSE  logical-name[:]

Command Qualifiers                Defaults

/ERROR=label                      None.
```

Prompts

Log_Name: logical-name[:]

Command Parameters

logical-name[:]

Specifies the logical name to be 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 explicitly closed with the CLOSE command, or until the process is deleted at logout. 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.

For a description of VAX/VMS file handling commands, see the VAX/VMS Guide to Using Command Procedures.

Command Qualifiers

/ERROR=label

Specifies a label on a line in the command procedure to receive control if the close request results in an error. If no error label is specified and an error occurs during the closing of the file, the command procedure continues execution at the next line in the file, as it does if no error occurs.

The error routine specified for this qualifier takes precedence over any action statement indicated in an ON command. 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 contains a successful completion value.

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 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
^Y
$ STOP
$ SHOW LOGICAL/PROCESS
   .
   .
   .

  INFILE = _DB1
  OUTFILE = _DB1
$ CLOSE INFILE
$ CLOSE OUTFILE
```

CTRL/Y interrupts the execution of the command procedure READFILE.COM and the STOP command stops it. 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.

COBOL/C74

Invokes the VAX-11 COBOL-74[1] compiler to compile a COBOL source program. The /C74 qualifier is required.

Format

```
COBOL/C74  file-spec[,...]

File Qualifiers                Defaults

/[NO]ANSI_FORMAT               /NOANSI_FORMAT
/[NO]COPY_LIST                 /COPY_LIST
/[NO]CROSS_REFERENCE           /NOCROSS_REFERENCE
/[NO]DEBUG[=option]            /DEBUG=TRACEBACK
/[NO]LIST[=file-spec]          (see text)
/[NO]MAP                       /NOMAP
/[NO]OBJECT[=file-spec]        (see text)
/[NO]VERB_LOCATION             /NOVERB_LOCATION
/[NO]WARNINGS                  /WARNINGS
```

Prompts

File: file-spec[,...]

Command Parameters

file-spec[,...]

Specifies one or more COBOL source programs to be compiled. If a file specification does not contain a file type, the compiler uses the default file type of COB.

You can specify more than one input file; each file is compiled separately. When you specify more than one input file, always separate the file specifications with commas (,). Do not use plus signs (+) as separators.

No wild card characters are allowed in the file specification.

1. Available under separate license.

File Qualifiers

/ANSI_FORMAT
/NOANSI_FORMAT

Indicates that the source program is in conventional ANSI format. The compiler then expects 80-character card image records with optional sequence numbers in character positions 1 through 6, indicators in position 7, Area A beginning in position 8, Area B beginning in position 12, and the identification area in positions 73 through 80.

The default is /NOANSI_FORMAT; that is, the compiler assumes that the source program is in DIGITAL's Terminal format, where Area A begins in column 1 and the source program records do not have line numbers. (Note that line numbers generated by editors are not part of the source program records.)

/COPY_LIST
/NOCOPY_LIST

Controls whether text copied from library files is printed in the listing file. If /NOCOPY_LIST is specified, only the COPY statement appears in the listing.

By default, the compiler includes all lines from files copied through the use of the COPY statement.

/CROSS_REFERENCE
/NOCROSS_REFERENCE

Controls whether the compiler creates a cross-referenced listing as part of the listing file. Data-names and procedure-names are listed in ascending order with the source program line numbers on which they appear. On the listing, the symbol # indicates the source line on which the name is defined.

By default, the compiler does not create a cross-referenced listing. Note that the /CROSS_REFERENCE qualifier significantly increases the compilation time for large source programs.

/DEBUG[=option]
/NODEBUG

Controls whether the compiler makes local symbol table and traceback information available to the debugger and the run time error reporting mechanism.

You can request one of the following options:

TRACEBACK	Provides the debugger with compiler-generated line numbers so that the debugger and error reporting mechanisms can translate virtual addresses into source program subroutine names and line numbers.
ALL	Provides traceback and local symbol table information. Note that /DEBUG=ALL is equivalent to /DEBUG.
NONE	Omits traceback and local symbol table information. Note that /DEBUG=NONE is equivalent to /NODEBUG.

By default, the compiler produces traceback information.

For details on how to debug a VAX-11 COBOL-74 program with the VAX-11 Symbolic Debugger, see the VAX-11 COBOL-74 User's Guide.

/LIST[=file-spec]
/NOLIST

Controls whether the compiler creates a listing file.

If you issue the COBOL/C74 command from interactive mode, the compiler, by default, does not create a listing file. If the COBOL/C74 command is executed from batch mode, /LIST is the default; the compiler gives a listing file the same file name as the input source file and a file type of LIS.

When you specify /LIST, you can control the defaults applied to the output file specification by the placement of the qualifier in the command, as described in Section 5.3.3, "Rules for Entering Output File Qualifiers." The compiler uses the default file type of LIS.

No wild card characters are allowed in the file specification.

/MAP
/NOMAP

Controls whether the compiler produces the following maps in the listing file:

- Data Division
- Procedure Map
- External Subprograms Referenced
- Data and Control PSECTs
- OTS Routines Referenced

By default, the compiler does not include these maps in the listing.

/OBJECT[=file-spec]
/NOOBJECT

Controls whether the compiler creates an output object module.

By default, the compiler produces an object module with the same file name as the source file and a file type of OBJ. When you specify /OBJECT, you can control the defaults applied to the output file specification by the placement of the qualifier in the command, as described in Section 5.3.3, "Rules for Entering Output File Qualifiers."

No wild card characters are allowed in the file specification.

/VERB_LOCATION
/NOVERB_LOCATION

Controls whether the compiler lists the object location for each verb in the source program. If you specify /VERB_LOCATION, the location appears on the line before the source line in which the verb is used.

By default, the compiler does not list the object location of verbs.

/WARNINGS
/NOWARNINGS

Controls whether the compiler prints informational diagnostic messages as well as warning and fatal diagnostic messages. By default, the compiler prints informational diagnostics; specify /NOWARNINGS to suppress them.

Examples

1.
```
$ COBOL/C74 TRANSLATE/LIST
$ LINK TRANSLATE,SYS$LIBRARY:C74LIB/LIBRARY
```

   The COBOL compiler compiles the source program TRANSLATE.COB and creates an object file named TRANSLATE.OBJ and a listing file named TRANSLATE.LIS.

   The LINK command specifies the object file, TRANSLATE.OBJ, and the COBOL-74 run-time library, C74LIB, that is located on the default system library device. This library is required to link all VAX-11 COBOL-74 images.

2.
```
$ COBOL/C74 READFILE/NOOBJECT/LIST-
$_/CROSS_REFERENCE/NOCOPY_LIST
```

   This command requests the compiler to create a listing file named READFILE.LIS, but no object file. The listing will contain a cross-reference listing, but will not contain any of the text copied from library files specified by COPY statements in the source program.

3.
```
$ ASSIGN "1,5" COB$SWITCHES
$ RUN COBTEST
```

   The ASSIGN command sets the program switch numbers 1 and 5 and resets all other switches. The RUN command executes the COBOL program COBTEST.EXE, which refers to these switches.

COBOL/RSX-11

Invokes the PDP-11 COBOL-74/VAX[1] compiler to compile a COBOL source program. The /RSX11 qualifier is required.

Format

```
COBOL/RSX11  file-spec[,...]
```

File Qualifiers	Defaults
/[NO]ANSI_FORMAT	/NOANSI_FORMAT
/[NO]COPY_LIST	/COPY_LIST
/[NO]CROSS_REFERENCE	/NOCROSS_REFERENCE
/[NO]LIST[=file-spec]	(see text)
/[NO]MAP	/NOMAP
/NAMES=aa	/NAMES=$C
/NEST=n	/NEST=10
/[NO]OBJECT[=file-spec]	(see text)
/[NO]OVERLAY	/NOOVERLAY
/SEGMENT_SIZE=n	
/[NO]VERB_LOCATION	/NOVERB_LOCATION
/[NO]WARNINGS	/NOWARNINGS

Prompts

File: file-spec[,...]

Command Parameters

file-spec[,...]

Specifies one or more COBOL source programs to be compiled. The file specification(s) must contain a file name; if you do not specify a file type, the compiler uses the default file type of CBL.

You can specify more than one input file; each file is compiled separately. When you specify more than one input file, always separate the file specifications with commas(,). Do not use plus signs (+) as separators.

No wild card characters are allowed in the file specification(s).

1. Available under separate license.

File Qualifiers

/ANSI_FORMAT
/NOANSI_FORMAT

Indicates whether the source program is in ANSI COBOL format or in DIGITAL's Terminal format.

An ANSI COBOL source file has 80-character records with Area A beginning in record position 8.

By default, the COBOL/RSX11 command assumes that the input records are in Terminal format, that is, Area A begins in record position 1 and the records do not have line numbers.

/COPY_LIST
/NOCOPY_LIST

Controls whether statements produced by COPY statements in the source program are printed in the listing file.

/COPY_LIST is the default: all source statements are included in the output listing.

/CROSS_REFERENCE
/NOCROSS_REFERENCE

Controls whether the compiler listing includes a cross reference listing. By default, the compiler does not create a cross reference listing.

/LIST[=file-spec]
/NOLIST

Controls whether the compiler produces an output listing and defines characteristics of the file.

If you issue the COBOL/RSX11 command from interactive mode, the compiler, by default, does not create a listing file. If you specify /LIST without a file specification, the compiler creates a listing with the same file name as the input file, but uses the file type of LST. If you include a file specification, the listing is written to that file or device.

If the COBOL/RSX11 command is executed from a batch job, /LIST is the default.

/MAP
/NOMAP

Requests the compiler to produce a Data Division map showing the memory addresses for Data Division entries.

The default is /NOMAP; the compiler does not produce a map.

/NAMES=aa

Requests the compiler to generate PSECT names starting with the 2-character prefix aa.

If the /NAMES qualifier is not specified, the default prefix of $C is used.

/NEST=n

Specifies the number of nested PERFORM statements allowed in the source program. By default, the compiler allows a maximum of 10 nested PERFORM statements.

/OBJECT[=file-spec]
/NOOBJECT

Controls whether the compiler produces an object file.

By default, the compiler produces an object file with the same file name as the input file and a file type of OBJ. The compiler also uses the default file type of OBJ when you include a file specification with the /OBJECT qualifier that does not have a file type.

No wild card characters are allowed in the file specification.

/OVERLAY
/NOOVERLAY

Controls whether the compiler makes procedural PSECTs overlayable.

By default, procedural PSECTS are not overlayable.

/SEGMENT_SIZE=n

Specifies the maximum size, in bytes, of procedure PSECTs created by the compiler. The minimum value allowed for n is 108 bytes.

Users must ensure that the absolute maximum task size of 65K bytes (including all PSECTs when linked) is not exceeded.

/VERB_LOCATION
/NOVERB_LOCATION

Indicates whether the output listing produced by the compiler shows the object location of each verb in the source program.

/WARNINGS
/NOWARNINGS

Controls whether the compiler prints informational diagnostic messages as well as warning and fatal diagnostic messages. By default, the compiler prints informational diagnostics; specify /NOWARNINGS to suppress them.

Examples

1.
```
$ COBOL/RSX11   MYFILE
```

   The COBOL command compiles the source statements in the file MYFILE.CBL and produces an object file named MYFILE.OBJ.

2.
```
$ COBOL/RSX11   TEST/OBJECT=TEST2/LIST
```

   The COBOL command compiles the source statements in the file TEST.CBL and produces an object file named TEST2.OBJ and a listing file named TEST.LST.

3.
```
$ COBOL/RSX11  SCANLINE
$ RUN SYS$SYSTEM:MRG
PLEASE ENTER FILE SPECIFICATION FOR OUTPUT FILE
SCAN.ODL
DO YOU WANT AN ABBREVIATED OR MERGED ODL FILE?
PLEASE ANSWER A(BBREVIATED) OR M(ERGED) A
DO YOU WANT TO OVERLAY I/O SUPPORT ROUTINES?
PLEASE ANSWER Y(ES) OR N(O) N
PLEASE ENTER FILE SPECIFICATION FOR INPUT ODL FILE
SCANLINE
OBJECT PROGRAM REFERENCED IN ODL FILE IS:
     SCANLINE.OBJ
PLEASE ENTER OBJECT FILE DEVICE AND UIC IN THE FORMAT: DEV:[GROUP,MEMBER]
 RET
ANY MORE INPUT ODL FILES?
PLEASE ANSWER Y(ES) OR N(O) N
ODL FILE MERGE COMPLETE
MERGED ODL FILE IS:
SCAN.ODL

CBL -- 15: STOP RUN

$ LINK/RSX11  SCAN/OVERLAY
$ RUN SCAN
```

   The COBOL/RSX11 command compiles the source program SCANLINE.CBL. The RUN command invokes the COBOL MERGE utility to create an overlay description file. The output from the MERGE utility, SCAN.ODL, is then specified as input to the LINK/RSX11 command. This command creates the executable image file, SCAN.EXE. The RUN command executes the image.

CONTINUE

Resumes execution of a DCL command, a program, or a command procedure that was interrupted by pressing CTRL/Y or CTRL/C. The CONTINUE command also serves as the target command of an IF or ON command in a command procedure, or following a label that is the target of a GOTO command. Additionally, the CONTINUE command can also resume a program that executed a VAX-11 FORTRAN PAUSE statement or a VAX-11 COBOL-74 STOP literal statement.

You can truncate the CONTINUE command to a single letter, C.

Format

CONTINUE

Command Qualifiers	Defaults
None.	None.

Prompts

None.

Command Parameters

None.

Description

After you interrupt an image, you cannot continue its execution if you have entered any command that executes another image. For a list of the commands that do not execute separate images, see Section 4.2.3, "Interrupting Program Execution."

You cannot continue execution of interrupted images that are privileged, that is, installed with privileges. See the description of the INSTALL Utility in the _VAX/VMS System Manager's Guide_ for more information on creating privileged images.

For more information on how to use commands like CONTINUE in command procedures, consult the _VAX/VMS Guide to Using Command Procedures_.

Examples

1.
```
$ RUN  MYPROGA
^Y
$ SHOW TIME
  18-JAN-1978 13:40:12
$ CONTINUE
```

The RUN command executes the program MYPROG. 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
```

This statement in a command procedure 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.

COPY

Creates a new file from one or more existing files. The COPY command can:

- Copy one file to another file
- Concatenate more than one file into a single output file
- Copy a group of files to another group of files

Format

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


Command Qualifiers          Defaults

/[NO]CONCATENATE            /CONCATENATE
/[NO]LOG                    /NOLOG


File Qualifiers             Defaults

/ALLOCATION=n
/[NO]CONTIGUOUS             /NOCONTIGUOUS
/EXTENSION=n
/FILE_MAXIMUM=n
/[NO]OVERLAY                /NOOVERLAY
/PROTECTION=code
/[NO]READ_CHECK             /NOREAD_CHECK
/[NO]REPLACE                /NOREPLACE
/[NO]TRUNCATE               /NOTRUNCATE
/VOLUME=n
/[NO]WRITE_CHECK            /NOWRITE_CHECK
```

Prompts

From: input-file-spec[,...]

To: output-file-spec

Command Parameters

input-file-spec[,...]

Specifies the names of one or more input files to be copied. If you specify more than one input file, you can separate them with either commas (,) or plus signs (+).

You can use full wild carding in the file specification(s), as described in Section 2.1.6.

output-file-spec

Specifies the name of the output file into which the input files are to be copied.

You must specify at least one field in the output file specification. If you do not specify a device and/or directory, the COPY command uses your current default device and directory. For other fields that you do not specify, the COPY command uses the corresponding field of the input file specification.

If you specify an asterisk (*) wild card character in place of the file name, file type, and/or version number field of the output file specification, the COPY command creates one or more output files, based on the input file specification. It uses the corresponding field of the first or related input file specification to name the output file.

Wild card characters are not allowed in the directory specification of an output file.

Description

When you specify more than one input file, the COPY command creates, by default, a single output file. You can specify multiple input files in any of the following ways:

- Separate input file specifications with commas (,) or plus signs (+).
- Specify wild card characters (Section 2.1.6) in place of the directory specification, file name, file type, and/or version number field of an input file specification.

The COPY command creates multiple output files when you specify multiple input files and one of the following:

- An asterisk (*) wild card character in the output directory specification, file name, file type, and/or version number field.
- Only a node name, a device name, or a directory specification in the output file specification.
- The /NOCONCATENATE qualifier.

When the COPY command creates multiple output files, it uses the corresponding field of each input file to name an output file.

When the COPY command creates a single output file for which any field of the output file specification contains an asterisk wild card character, the COPY command uses the corresponding field of the first, or only, input file to name the output file.

Use the /LOG qualifier when you specify multiple input and output files to verify the files actually were copied.

Version Numbers: If no version numbers are specified for input and output files, the COPY command, by default, gives the output file a version number of 1, or increments by 1 the version number of an existing file with the same file name and file type.

If the input or output file version number is explicit or a wild card character, the COPY command by default gives the output files the same version numbers as the associated input files. If an equal or higher version of the output file already exists, the COPY command issues a warning message and does not copy the 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 had 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.

Command Qualifiers

/CONCATENATE
/NOCONCATENATE

Controls, when a wild card character is used in any component of the output file specification, whether a single output file is to be created from all files that satisfy the input file specification.

By default, a wild card character in an input file specification results in a single output file consisting of the concatenation of all input files matching the file specification.

When you concatenate files from Files-11 Structure Level 2 disks, the COPY command concatenates the files in alphanumeric order; if you specify a wild card character in the file version field, files are copied in descending order by version number. When you concatenate files from Files-11 Structure Level 1 disks, the COPY command concatenates the files in random order.

/LOG
/NOLOG

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

If you specify /LOG, the COPY command displays, for each copy operation, the file specifications of the input and output files, 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 the total number of new files created.

File Qualifiers

/ALLOCATION=n

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

If not specified, the initial allocation of the output file is determined by the size of the input file being copied.

/CONTIGUOUS
/NOCONTIGUOUS

Indicates whether the output file is to be contiguous, that is, whether the file must occupy consecutive physical disk blocks.

By default, the COPY command creates an output file in the same format as the corresponding input file. If an input file is contiguous, the COPY command attempts to create a contiguous output file, but it does not report an error if there is not enough space. If you copy multiple input files of different formats, the output file may or may not be contiguous. 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 two COPY commands: once, to copy the file from the tape, and a second time to create a contiguous file.

/EXTENSION=n

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.

The owner UIC of the output file is the UIC of the current process.

/FILE_MAXIMUM=n

Specifies the maximum number of logical records that the output file can contain.

This qualifier is valid only for relative files. For information on creating and using relative files, see the VAX-11 Record Management Services Reference Manual.

/OVERLAY
/NOOVERLAY

Requests that data in the input file be copied into an existing output file, overlaying the existing data. 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

Defines the protection to be applied to the output file.

Specify the protection code using the rules given in Section 5.10. Any protection attributes not specified are taken from the current protection of the corresponding input file.

The owner UIC of the output file is the UIC of the current process.

/READ_CHECK
/NOREAD_CHECK

Requests the COPY command to read each record in the specified input file(s) twice to verify that all records were correctly read.

By default, records are not read twice.

/REPLACE
/NOREPLACE

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.

By default, the COPY command creates a new version of a file if the file already exists, incrementing the version number.

/TRUNCATE
/NOTRUNCATE

Controls whether the COPY command truncates an output file at the end-of-file when copying it. By default, the COPY command uses the allocation of the input file to determine the size of the output file.

/VOLUME=n

Requests that the COPY command place the entire output file on the specified relative volume number of a multivolume set.

If the /VOLUME qualifier is not specified, the file is placed in an arbitrary position within the multivolume set.

/WRITE_CHECK
/NOWRITE_CHECK

Requests the COPY command to read 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.

By default, the output records are not read after writing.

Examples

1.
```
$ COPY   TEST.DAT   NEWTEST.DAT
```

The COPY command copies the contents of the file TEST.DAT from the default disk and directory into a file named NEWTEST.DAT. 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
```

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/REPLACE  TEST.DAT  NEW.DAT;1
%COPY-I-REPLACED, DBA2:[MAL]NEW.DAT;1 being replaced
%COPY-S-COPIED, DBA2:[MAL]TEST.DAT;1 copied to DBA2:[MAL]NEW.DAT;1 (1 block)
%COPY-S-NEWFILES, 1 file created
```

The /REPLACE qualifier requests the COPY command to 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]
```

The COPY command copies the highest versions of files in the current default directory with a file type of COM to the subdirectory MALCOLM.TESTFILES.

5.
```
$ COPY/LOG  *.TXT   *.OLD
%COPY-S-COPIED, DBA2:[MAL]A.TXT;2 copied to DBA2:[MAL]A.OLD;2 (1 block)
%COPY-S-COPIED, DBA2:[MAL]B.TXT;2 copied to DBA2:[MAL]B.OLD;2 (1 block)
%COPY-S-COPIED, DBA2:[MAL]G.TXT;2 copied to DBA2:[MAL]G.OLD;2 (4 blocks)
%COPY-S-NEWFILES, 3 files created
```

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 it created.

6.
```
$ COPY/LOG  A.DAT,B.MEM C.*
%COPY-S-COPIED, DBA2:[MAL]A.DAT;5 copied to DBA2:[MAL]C.DAT;11 (1 block)
%COPY-S-COPIED, DBA2:[MAL]B.MEM;2 copied to DBA2:[MAL]C.MEM;24 (58 records)
%COPY-S-NEWFILES, 2 files created
```

The input file specifications are separated with a comma to request that two files be copied. The asterisk wild card 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, DBA2:[MAL]A.TXT;2 copied to DBA2:[MAL]TXT.SAV;1 (1 block)
%COPY-S-APPENDED, DBA2:[MAL]B.TXT;2 appended to DBA2:[MAL]TXT.SAV;1 (3 records)
%COPY-S-APPENDED, DBA2:[MAL]G.TXT;2 appended to DBA2:[MAL]TXT.SAV;1 (51 records)
%COPY-S-NEWFILES, 1 file created
```

The COPY command copies the highest versions of all files with file types of 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 specify /NOCONCATENATE in this example, the COPY command creates multiple versions of the file TXT.SAV.

8.
```
$ COPY MASTER.DOC DMA1:[BACKUP]
```

The COPY command copies the highest version of the file MASTER.DOC to the device DMA1. If no file named MASTER.DOC already exists in the directory BACKUP, the COPY command uses the version number of the input file.

9.
```
$ MOUNT   MTA1:   VOL025   TAPE:
$ COPY  TAPE:*.*   *
```

The MOUNT command requests that the volume labeled VOL025 be mounted on the magnetic tape device MTA1 and assigns the logical name TAPE to the device.

The COPY command uses the logical name TAPE for 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.

10.
```
$ ALLOCATE CR:
  _CRA0:  ALLOCATED
$ COPY CRA0:  CARDS.DAT
$ DEALLOCATE CRA0:
```

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, CRA0.

After the card reader is allocated, you can place a deck of cards in the reader and issue 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.

CORAL

Invokes the VAX-11 CORAL 66[1] compiler to compile one or more CORAL 66 source programs. For more information on the VAX-11 CORAL 66 compiler, see the IAS/RSX/VMS CORAL 66 User's Guide.

Format

```
CORAL  file-spec[,...]

Command Qualifiers                 Defaults

None.                              None.

File Qualifiers                    Defaults

/[NO]CHECK                         /NOCHECK
/[NO]IECCA                         /NOIECCA
/[NO]LIST[=file-spec]              (see text)
/[NO]OBJECT[=file-spec]            (see text)
/[NO]SHOW[=(option[,...])]         /SHOW=(SOURCE,SYMBOLS,-
                                      STATISTICS)
/TEST=n                            /TEST=0
/WIDTH=n                           /WIDTH=132
```

Prompts

File: file-spec[,...]

Command Parameters

file-spec[,...]

Specifies one or more CORAL 66 language source programs to be compiled. If you do not specify a file type for an input file, the compiler uses the default file type of COR.

You can specify more than one input file. If you separate the file specifications with commas (,), each file is compiled separately. If you separate the file specifications with plus signs (+), the files are concatenated and compiled as a single input file, producing single object and listing files.

No wild card characters are allowed in the file specification(s).

1. Available under separate license.

File Qualifiers

/CHECK
/NOCHECK

Indicates whether the compiler should generate additional code at each reference to an array or table element. Such code permits each reference to be checked at run time to ensure it lies within the declared bounds.

By default, no checking is done.

/IECCA
/NOIECCA

Defines whether the compiler highlights non-IECCA keywords in the listing file with warning messages.

By default, no highlighting occurs.

/LIST[=file-spec]
/NOLIST

Indicates whether an output listing is created, and optionally provides an output file specification for the listing file.

If you issue the CORAL command interactively, the compiler, by default, does not create a listing file. When /NOLIST is present, either explicitly or by default, errors are reported on the current output device.

If you execute the CORAL command in a batch job, the /LIST qualifier is the default. When you specify /LIST, you can control the defaults applied to the output file specification by the placement of the qualifier in the command, as described in Section 5.3.3, "Rules for Entering Output File Qualifiers." The default file type provided for listing files is LIS.

No wild card characters are allowed in the file specification.

/OBJECT=file-spec
/NOOBJECT

Controls whether an object module is created by the assembler. It also defines the file specification for the file.

By default, the compiler creates an object module with the same file name as the first input file and file type of OBJ. When you specify /OBJECT, you can control the defaults applied to the output file specification by the placement of the qualifier in the command, as described in Section 5.3.3, "Rules for Entering Output File Qualifiers." The default file type provided for object files is OBJ.

No wild card characters are allowed in the file specification.

/SHOW[=(option[,...])]
/NOSHOW[=(option[,...])]

Controls the contents of the source listing file. You may specify any of the following options:

MACROS	Outputs the macro expansions and the source listing
OVERRIDE	Overrides any NOLIST statements in the source code
SOURCE	Outputs the source listing
STATISTICS	Outputs compilation statistics
SYMBOLS	Outputs the symbol table list

If you specify more than one option, separate them by commas and enclose the list in parentheses.

By default, source listings with symbol tables and compilation statistics are produced.

/TEST=n

Retains or omits certain declarations and statements from the compilation without requiring special editing of the source text. Only those VAX-11 CORAL 66 statements or declarations that carry the TEST keyword with values less than or equal to the value given as n are compiled.

By default, all statements and declarations are not compiled if you omit the /TEST qualifier.

/WIDTH=n

Controls the width of the listing file. You may specify any decimal number between 8 and 132 for the width n.

By default, the width of the listing is 132 columns if you omit the /WIDTH qualifier.

Examples

1. $ **CORAL FRED/TEST=**8

   Compiles all CORAL 66 statements in FRED.COR that are preceded by 'TEST'x, where x must be less than or equal to eight.

2. $ **CORAL JIM + JOE /LIST=JIM/NOSHOW=(STATISTICS,MACROS)**

   The input source files JIM.COR and JOE.COR are concatenated before compilation to produce the object module JIM.OBJ and the listing file JIM.LIS. The printing of the compilation statistics and macro expansions is suppressed in JIM.LIS.

CREATE

Creates one or more sequential disk files from records that follow the command in the input stream.

Format

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


Command Qualifiers        Defaults

/[NO]LOG                  /NOLOG
/OWNER_UIC=uic
/PROTECTION=code
/VOLUME=n
```

Prompts

File: file-spec[,...]

Command Parameters

file-spec[,...]

Specifies the name of one or more input files to be created.

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 you do not specify a file version number, and a file already exists with the same file name and file type as the file specification, the CREATE command creates a new version of the file.

No wild card characters are allowed in the file specifications.

Command Qualifiers

/LOG
/NOLOG

Controls whether the CREATE command displays the file specification of each file that it has created.

By default, the CREATE command does not display the names of files after it creates them.

/OWNER_UIC=uic

Specifies the user identification code to be associated with the file being created. Specify the UIC in the format:

[g,m]

g	is an octal number in the range 0 through 377 representing the group number.
m	is an octal number in the range 0 through 377 representing the member number.

The bold brackets ([ ]) are required in the UIC specification.

If you do not specify an owner UIC when you create a file, the command assigns your UIC to the file.

Note, you must have the SYSPRV user privilege to specify a UIC other than your own UIC.

/PROTECTION=code

Defines the protection to be applied to the file. Specify the protection code according to the rules given in Section 5.10. Any categories not specified are denied all types of access.

If you do not specify a protection code when you create a file, the command applies your current default protection to the file.

/VOLUME=n

Requests that each file be placed on the specified relative volume number of a multivolume set.

If you omit the /VOLUME qualifier, files are placed in arbitrary positions within the multivolume set.

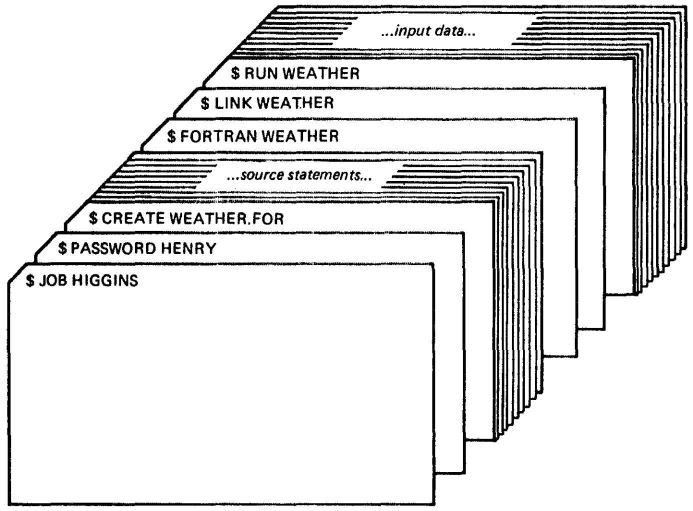
Examples

1.
```
$ CREATE   A.DAT,B.DAT
Input line one for A.DAT...
Input line two for A.DAT...
    .
    .
    .

^Z
Input line one for B.DAT...
Input line two for B.DAT...
    .
    .
    .
^Z
$
```

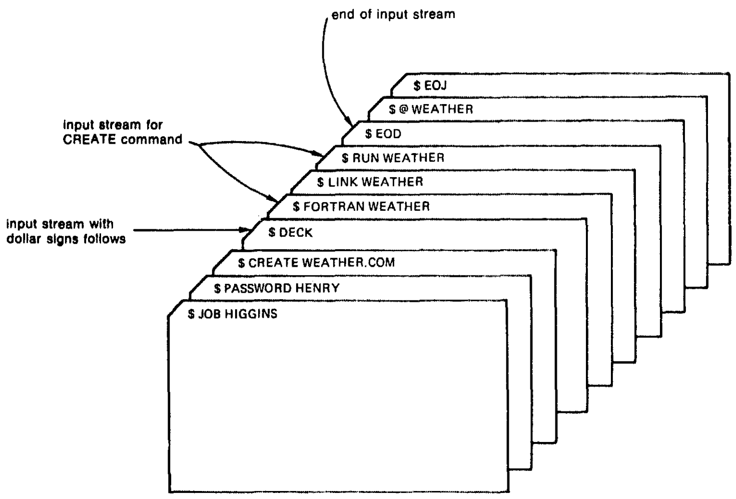
After you issue the CREATE command from the terminal, the system reads input lines into the sequential file A.DAT until CTRL/Z terminates the first input. The next set of input data is placed in the second file, B.DAT.

2.



When you issue 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. In this batch job example, the CREATE command creates a FORTRAN source file. The next commands compile, link, and run the file just created. Input data follows the RUN command.

3.



This batch job example illustrates using 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 procedure is executed with the @ (Execute Procedure) command.

4.
```
$ CREATE AAA.DAT/LOG
   input data
   .
   .
   .
^Z
%CREATE-I-CREATED, DMA0:[MALCOLM]AAA.DAT;1 created
$
```

This CREATE command illustrates the effect of the /LOG qualifier. Once the file is successfully created, a message appears identifying the file by device and directory, file name, file type, and version number.

CREATE/DIRECTORY

Defines a new directory or subdirectory for cataloging files. The /DIRECTORY qualifier is required.

Format

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

    Additional
Command Qualifiers                   Defaults

/[NO]LOG                             /NOLOG
/OWNER_UIC=uic
/PROTECTION=code
/VERSION_LIMIT=n
/VOLUME=n
```

Prompts

File: directory-spec[,...]

Command Parameters

directory-spec[,...]

Specifies the name of one or more directories or subdirectories to be created.

The directory specifications must contain a directory name, and optionally can contain a device name. When you create a subdirectory, separate the names of the directory levels with periods (.).

No wild card characters are allowed in the directory specification.

Description

To create a first-level directory you must be allowed write access to the master file directory on the volume on which you are creating the directory. On a system volume, normally only users with a system UIC or the SYSPRV or BYPASS user privileges are allowed write access to the MFD to create a first level directory. To create a subdirectory, you must be allowed write access to the lowest level directory that currently exists.

Additional Command Qualifiers

/LOG
/NOLOG

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

By default, the CREATE/DIRECTORY command does not display the name of each directory after it creates it.

/OWNER_UIC=uic

Specifies the user identification code to be associated with the directory being created. Specify the UIC in the format:

[g,m**]**

g	is an octal number in the range 0 through 377 representing the group number.
m	is an octal number in the range 0 through 377 representing the member number.

The bold brackets (**[ ]**) are required in the UIC specification.

If you do not specify the /OWNER_UIC qualifier when you create a directory, the command assigns ownerships as follows:

- If you specify the directory name in either alphanumeric or subdirectory format, ownership defaults to your UIC
- If you specify the directory name in UIC format (Section 2.1.3.1), ownership defaults to the UIC in the directory name

/PROTECTION=code

Defines the protection to be applied to the directory. Specify the protection code according to the rules given in Section 5.10. Any categories not specified are denied all types of access.

If you do not specify the /PROTECTION qualifier when you create a directory, the command uses the protection in effect for the next-highest level directory, less any delete access. If you are creating a first-level directory, then the protection of the MFD is used. (The protection of the MFD is established by the INITIALIZE command.)

/VERSION_LIMIT=n

Specifies that no more than n versions of each file created in this directory are to be kept. Whenever n versions exist and a new version is created, the lowest version is automatically deleted. If you omit the /VERSION_LIMIT qualifier, the default is the number of versions permitted for the directory at the next-higher level.

You may specify /VERSION_LIMIT=0. This creates a directory with no version limit.

Regardless of what version limit you assign to the directory with this qualifier, the system limits the number of existing versions of any file to approximately 60. The upper limit prevails even when the version limit has been set to a value greater than 60, or when the version limit is set to zero.

/VOLUME=n

Requests that the directory file be placed on the specified relative volume number of a multivolume set.

If you omit the /VOLUME qualifier, the file is placed in an arbitrary position within the multivolume set.

Examples

1.
```
$ CREATE/DIRECTORY DMA2:[MALCOLM]
```

The CREATE/DIRECTORY command creates a directory named MALCOLM on the device DMA2.

2.
```
$ CREATE/DIRECTORY [MALCOLM.SUB]
$ SET DEFAULT [MALCOLM.SUB]
```

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

3.
```
$ CREATE/DIRECTORY/PROTECTION=(SYSTEM:RWED,OWNER:RWED,GROUP,WORLD) -
$_[MALCOLM.SUB.HLP]
```

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.

DEALLOCATE

Returns a device that was reserved for private use to the pool of available devices in the system.

Format

```
DEALLOCATE    [device-name[:]]
```

Command Qualifiers	Defaults
/ALL	None.

Prompts

Device: device-name[:]

Command Parameters

device-name[:]

Specifies the name of the device to be deallocated. The device name can be a physical device name or a logical name.

If you omit the controller designator and/or unit number, they default to controller A and unit 0, respectively.

Command Qualifiers

/ALL

Requests that all devices you have currently allocated be deallocated.

If you specify /ALL, you cannot specify a device name.

Examples

1. **$ DEALLOCATE _DMB1:**

   The DEALLOCATE command deallocates unit 1 of the RK06 device(s) on controller B. The underscore character in the device name indicates that it is a physical device name; the DEALLOCATE command does not check to see if it is a logical name.

2.
```
$ ALLOCATE   MT:   TAPE:
  _MTB1:  ALLOCATED
   .
   .
   .

$ DEALLOCATE  TAPE
```

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.

Note that a colon was specified on the logical name TAPE in the ALLOCATE command, but that the colon can be omitted on the DEALLOCATE command.

3.
```
$ DEALLOCATE/ALL
```

The DEALLOCATE command deallocates all devices that are currently allocated.

DEASSIGN

Cancels logical name assignments made with the ASSIGN, DEFINE, or ALLOCATE commands.

Format

```
DEASSIGN   [logical-name[:]]
```

Command Qualifiers	Defaults
/ALL	
/GROUP	/PROCESS
/PROCESS	/PROCESS
/SUPERVISOR_MODE	/SUPERVISOR_MODE
/SYSTEM	/PROCESS
/USER_MODE	/SUPERVISOR_MODE

Prompts

Log_Name: logical-name[:]

Command Parameters

logical-name[:]

Specifies a 1- through 63-character logical name to be deassigned. If the logical name contains any characters other than alphanumeric, dollar sign ($), or underscore (_) characters, enclose it in quotation marks (").

If you terminate the logical-name parameter with a colon (:), the command interpreter ignores it. (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 actual logical name, you must specify two colons on the logical-name parameter for the DEASSIGN command.

The logical-name parameter is required unless you specify /ALL.

Description

If you enter more than one of the qualifiers /PROCESS, /GROUP or /SYSTEM, 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 only from the specified logical name table.

The command interpreter deassigns all supervisor mode entries in the process logical name table when you log off the system. User mode entries are deassigned when any image exits. Names in the group or system logical name tables must be explicitly deassigned.

Command Qualifiers

/ALL

Specifies that all logical names in the specified logical name table are to be deleted. If no logical name table is specified, all process logical name table entries are deleted.

If you specify /ALL, you cannot enter a logical-name parameter.

/GROUP

Indicates that the specified logical name is in the group logical name table.

The user privilege GRPNAM is required to delete entries from the group logical name table.

/PROCESS

Indicates that the specified logical name is in the process logical name table. This is the default.

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, you can deassign the names you explicitly created.

/SUPERVISOR_MODE

Indicates, for entries in the process logical name table, that an entry exists in supervisor mode. This is the default; /SUPERVISOR_MODE deletes both user and supervisor mode entries.

/SYSTEM

Indicates that the specified logical name is in the system logical name table.

The user privilege SYSNAM is required to delete entries from the system logical name table.

/USER_MODE

Indicates, for entries in the process logical name table, that the entry exists in user mode. /USER_MODE deletes only user mode entries.

Examples

1.
```
$ SHOW LOGICAL TEST_CASES
  TEST_CASES = DBA1:[HARVEY]FILES.DAT (process)
$ DEASSIGN   TEST_CASES
$ SHOW LOGICAL TEST_CASES
  No translation for logical name TEST_CASES
```

   The SHOW LOGICAL command displays the current equivalence name for the logical name TEST_CASES. The DEASSIGN command deassigns the equivalence name; the next SHOW LOGICAL command indicates that the name is deassigned.

2.
```
$ ASSIGN DBA1:   COPY:
$ DEASSIGN COPY
```

   The ASSIGN command equates the logical name COPY with the device DBA1 and places the names in the process logical name table. The DEASSIGN command deletes the logical name. Note that a colon was specified on the logical name COPY in the ASSIGN command, but that the colon can be omitted on the DEASSIGN command.

3.
```
$ DEFINE SWITCH:  TEMP
$ DEASSIGN SWITCH::
```

   The DEFINE command places the logical name SWITCH: in the process logical name table. Two colons are required on the DEASSIGN command to delete this logical name because the DEFINE command does not remove trailing colons from logical names.

4.
```
$ ASSIGN/GROUP _DBB2: GROUP_DISK
$ DEASSIGN/PROCESS/GROUP GROUP_DISK
```

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

5.
```
$ DEASSIGN/ALL
```

   The DEASSIGN command deletes all names from the process logical name table. This command does not, however, delete the names that were placed in the process logical name table in executive mode by the command interpreter (SYS$INPUT, SYS$OUTPUT, SYS$ERROR, SYS$DISK, and SYS$COMMAND).

DEBUG

Invokes the VAX-11 Symbolic Debugger after program execution is interrupted by CTRL/C or CTRL/Y.

Format

```
DEBUG
```

Command Qualifiers	Defaults
None.	None.

Prompts

DBG>

Command Parameters

None.

Description

When a program image is executing, it can be interrupted by CTRL/C or CTRL/Y. Following the interruption, the DEBUG command can be issued to pass control to the debugger; this function is useful when a program is in an infinite loop and you want to gain control and use the debugger to determine the cause of the problem.

If no image is currently executing, the DEBUG command performs no operation.

You need not specify the /DEBUG qualifier in the LINK command to invoke the debugger. However, images linked with /NODEBUG, contain limited symbolic information. If you specify the /NOTRACEBACK qualifier in a LINK command, any subsequent DEBUG command causes a software exception condition. If the image has not declared a condition handler, this exception condition may cause the termination of the image.

For complete details on the commands available to debug programs, see the VAX-11 Symbolic Debugger Reference Manual.

Examples

1.
```
$ FORTRAN/DEBUG/NOOPTIMIZE   WIDGET
$ LINK/DEBUG  WIDGET
$ RUN WIDGET

              VAX-11 DEBUG V2.0

%DEBUG-I-INITIAL, language is FORTRAN, module set to 'WIDGET'
DBG>GO
ENTER NAME:
ENTER NAME:
ENTER NAME:
^Y
$ DEBUG
DBG>
```

The FORTRAN and LINK commands both specify the /DEBUG qualifier, to compile the program WIDGET.FOR with debugger symbol table information and to include the debugger in the image file. The RUN command begins execution of the image WIDGET.EXE, which loops uncontrollably. CTRL/Y interrupts the program, and the DEBUG command gives control to the debugger.

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 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.

Format

```
    $ DECK

    Command Qualifiers          Defaults

    /DOLLARS[=string]           /DOLLARS=$EOD
```

Prompts

None.

Command Parameters

None.

Description

The DECK command defines an end-of-file indicator only for a single data stream; it allows 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, each 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, any record beginning with a dollar sign ($). The default is also reset if an actual end-of-file occurs for the current command level.

The DECK command is invalid if it is not preceded by a request to execute a command or program that requires input data.

For more information on how to use commands like this one in command procedures, consult the VAX/VMS Guide to Using Command Procedures.

Command Qualifiers

/DOLLARS[=string]

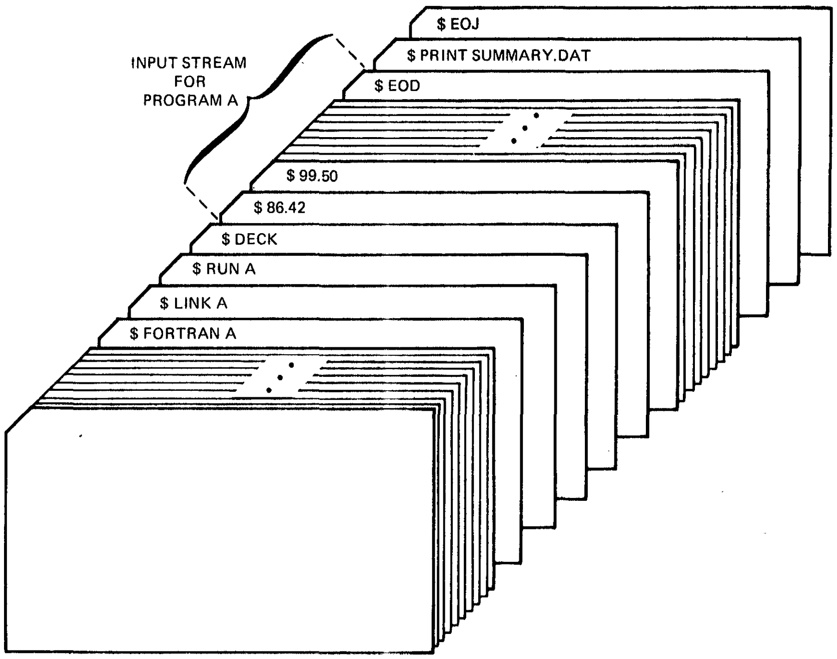
Sets the end-of-file indicator to the specified string.

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.

Specify a string if the input data contains one or more records beginning with the string $EOD. The string can have from 1 through 15 characters. Enclose it in quotation marks (") if you want to specify an end-of-file indicator that contains literal lowercase letters or multiple blanks or tabs. The command interpreter does not scan the string; thus, no symbol substitution can be performed.

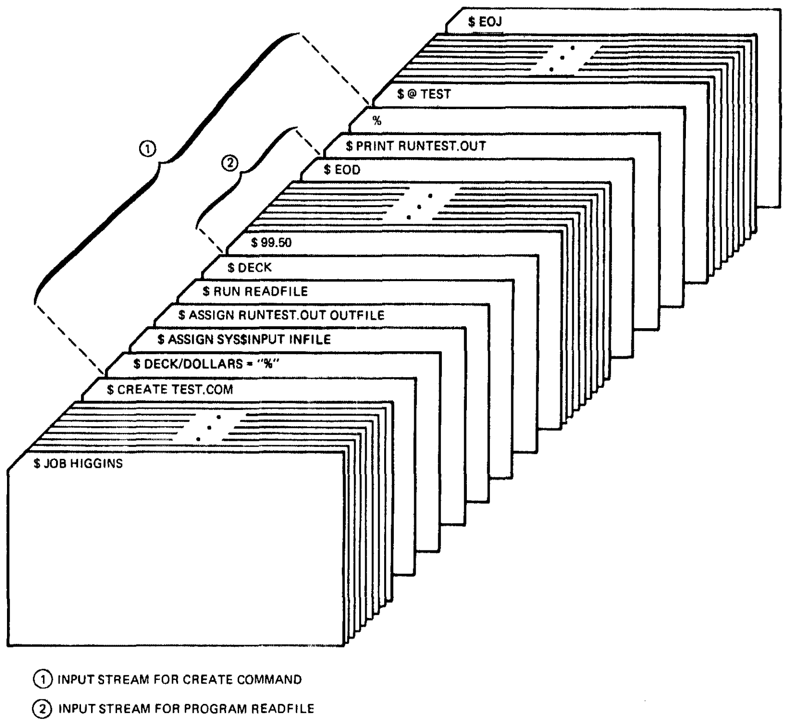
Examples

1.



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 may contain dollar signs. The EOD command signals end-of-file for the data.

2.


$ EOJ
$ @ TEST
%
$ PRINT RUNTEST.OUT
$ EOD
$ 99.50
$ DECK
$ RUN READFILE
$ ASSIGN RUNTEST.OUT OUTFILE
$ ASSIGN SYS$INPUT INFILE
$ DECK/DOLLARS = "%"
$ CREATE TEST.COM
$ JOB HIGGINS
①
②
① INPUT STREAM FOR CREATE COMMAND
② INPUT STREAM FOR PROGRAM READFILE

The CREATE command 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.

Creates a logical name table entry and assigns an equivalence name string to the specified logical name. The DEFINE command is similar in function to the ASSIGN command; however, its primary purpose is to assign logicalname/equivalencename pairs for application-specific uses other than for logical file specification assignments.

Format

```
DEFINE   logical-name[:]  equivalence-name[:]
```

Command Qualifiers	Defaults
/GROUP	/PROCESS
/PROCESS	/PROCESS
/SUPERVISOR_MODE	/SUPERVISOR_MODE
/SYSTEM	/PROCESS
/USER_MODE	/SUPERVISOR_MODE

Prompts

Log_Name:	logical-name[:]
Equ_Name:	equivalence-name[:]

Command Parameters

logical-name[:]

Specifies a 1- through 63-character logical name string. If the string contains any characters besides alphanumeric or underscore characters, enclose it in quotation marks (").

equivalence-name[:]

Defines the 1- through 63-character equivalence name to be associated with the logical name in the specified logical name table. If the string contains other than alphanumeric or underscore characters, it must be enclosed in double quotation marks (").

Description

If you enter more than one of the qualifiers /PROCESS, /GROUP, or /SYSTEM, or both of the qualifiers /SUPERVISOR_MODE and /USER_MODE, only the last qualifier entered is accepted.

For additional information on using logical names, see Section 2.2, "Logical Names."

Command Qualifiers

/GROUP

Places the logical name/equivalence name pair 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 user privilege GRPNAM is required to place a name in the group logical name table.

/PROCESS

Places the logical name/equivalence name pair in the process logical name table. This is the default.

/SUPERVISOR_MODE

Specifies, for an entry in the process logical name table, that the logical name be entered in supervisor mode.

This is the default for process logical name table entries. The /SUPERVISOR_MODE qualifier is ignored when entries are made in the group or system logical name tables.

/SYSTEM

Places the logical name/equivalence name pair in the system logical name table. All system users can access the logical name.

The user privilege SYSNAM is required to place a name in the system logical name table.

/USER_MODE

Specifies, for an entry in the process logical name table, that the logical name be entered in user mode.

A user mode logical name is practical for the execution of a single image; it allows an image executing in a command procedure to redefine SYS$INPUT. User mode entries are deleted when any image executing in the process exits (that is, after any DCL command or user program that executes an image completes execution), or when a STOP command is issued.

By default, process logical name table entries are made in supervisor mode. The /USER_MODE qualifier is ignored when entries are made in the group or system logical name tables.

Examples

1.
```
$ DEFINE PROCESS_NAME LIBRA
$ RUN WAKE
```

   The DEFINE command places the logical name PROCESS_NAME in the process logical name table with an equivalence name of LIBRA. The program WAKE can translate the logical name PROCESS_NAME to perform some special action on the process named LIBRA.

2.
```
$ DEFINE TEMP: DBA1:
$ DEASSIGN TEMP::
```

   The DEFINE command creates an equivalence name for the logical name TEMP: and places the name in the process logical name table. The DEASSIGN command deletes the logical name. Note that two colons are required on the logical name in the DEASSIGN command because the DEFINE command does not remove trailing colons from logical names, as the DEASSIGN command does.

DELETE

Deletes one or more files from a mass storage disk volume.

Format

```
DELETE    file-spec[,...]

Command Qualifiers        Defaults

/BEFORE[=time]
/[NO]CONFIRM              /NOCONFIRM
/CREATED
/EXPIRED
/[NO]LOG                  /NOLOG
/MODIFIED
/SINCE[=time]
```

Prompts

File: file-spec[,...]

Command Parameters

file-spec[,...]

Specifies the names of one or more files to be deleted. The first file specification must contain an explicit or default directory specification plus a file name, a file type, and a version number; subsequent file specifications must contain a version number. You can specify wild card characters in any of the file specification fields (see Section 2.1.6).

A semicolon followed by no file version number, a version number of 0, or one or more blanks in the version number of a file specification results in the deletion of the latest version of the file.

To delete more than one file, separate the file specifications with commas (,) or plus signs (+).

If you omit the directory specification or device name, the current default device and directory are used.

Command Qualifiers

/BEFORE[=time]

Specifies that only the files dated earlier than a particular time be deleted. You can specify an absolute date and time. Use the syntax rules for date and time values specified in Section 5.8.

If you specify /BEFORE and do not specify a value, the DELETE command assumes /BEFORE=TODAY.

Use the /CREATED, /EXPIRED, or /MODIFIED qualifiers to request that only files created, expired, or modified before the specified time be deleted. If none of these qualifiers is specified, the DELETE command deletes all files created and modified within the specified time.

/CONFIRM
/NOCONFIRM

Controls whether the DELETE command displays the file specification of each file before deleting and requests you to confirm whether or not the file should actually be deleted. If you specify /CONFIRM, you must respond to a prompt with a Y, followed by a carriage return, before the DELETE command will delete the file. If you enter anything else, the file is not deleted.

By default, the DELETE command does not request confirmation of files it is deleting.

/CREATED

Specifies, when /BEFORE and/or /SINCE is specified, that only files created within the defined time period be deleted.

The default is /MODIFIED if none of the qualifiers /CREATED, /MODIFIED, or /EXPIRED is specified.

/EXPIRED

Specifies, when /BEFORE and/or /SINCE is specified, that only files that reached their expiration dates within the specified time be deleted.

If any file does not have an expiration date associated with it, it is assumed to have expired at the time the DELETE command is issued. (Files can be assigned expiration dates when they are created or revised with RMS.)

/LOG
/NOLOG

Controls whether the DELETE command displays the file specification of each file after its deletion.

By default, the DELETE command does not display the names of files after it deletes them.

/MODIFIED

Specifies, when /BEFORE and/or /SINCE are specified, that only files that were modified within the defined time period be deleted. A file's revision date is updated whenever the file is accessed and updated.

/SINCE[=time]

Specifies that only the files dated later than a particular time be deleted. You can specify an absolute date and time. Use the syntax rules for date and time values specified in Section 5.8.

If you specify /SINCE and do not specify a value, the DELETE command assumes /SINCE=TODAY.

Use the /CREATED, /EXPIRED, or /MODIFIED qualifiers to request that only files created, expired, or modified after the specified time be deleted. If none of these qualifiers is specified, the DELETE command deletes all files created and modified within the specified time.

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 types of 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 version numbers (here specified as wild cards) must be included in each file specification.

4.
```
$DELETE *.DAT;*/BEFORE=01-JUN/LOG
%DELETE-I-DELETED, DBA2:[MALCOLM]ASSIGN.DAT;1 deleted
%DELETE-I-DELETED, DBA2:[MALCOLM]BATCHAVE.DAT;1 deleted
%DELETE-I-DELETED, DBA2:[MALCOLM]CANCEL.DAT;1 deleted
%DELETE-I-DELETED, DBA2:[MALCOLM]DEFINE.DAT;1 deleted
%DELETE-I-DELETED, DBA2:[MALCOLM]EXIT.DAT;1 deleted
```

   The DELETE command deletes all versions of all files with file types of DAT that were either created or updated before June 1, this year.

5.
```
$DELETE A.B;RET
```

   The DELETE command deletes the file A.B with the highest version number.

6.
```
$DELETE [MALCOLM.TESTFILES]*.OBJ;*/CONFIRM/SINCE=TODAY

DBA2:[MALCOLM.TESTFILES]AVERAG.OBJ;1, delete? (Y or N):Y
DBA2:[MALCOLM.TESTFILES]SCANLINE.OBJ;2, delete? (Y or N):N
DBA2:[MALCOLM.TESTFILES]WEATHER.OBJ;3, delete? (Y or N):Y
```

The DELETE command examines all versions of files with file types of 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.

7.
```
$DIRECTORY [.SUBTEST]
  No files found.
$SET PROTECTION SUBTEST.DIR/PROTECTION=OWNER:D
$DELETE SUBTEST.DIR;1
```

Before deleting the directory file SUBTEST.DIR, 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.

DELETE/ENTRY

Deletes one or more entries from a printer or batch job queue. The /ENTRY qualifier is required.

Format

```
DELETE/ENTRY=(job-number[,...])  queue-name[:]
```

Additional Command Qualifiers	Defaults
None.	None.

Prompts

```
Queue:  queue-name[:]
```

Command Parameters

job-number[,...]

Specifies the job number of the job to be deleted from the queue. The /ENTRY qualifier requires at least one job-number parameter to specify the job number(s) of one or more jobs to be deleted from a printer of batch job queue. If you specify more than one job number, separate them by commas and enclose the list in parentheses.

queue-name[:]

Specifies the name of the queue in which the job(s) exist.

Description

Unless you possess certain user privileges, the job(s) to be deleted must have been queued by your process. You can also delete any process in the same group as the current process, provided you have the GROUP user privilege. Otherwise, you need the WORLD or OPER user privileges to delete a process that is not your own or in your group.

You can delete jobs that have not yet begun processing or files that are currently being processed.

Examples

1.
```
$ PRINT/HOLD   ALPHA.TXT
  Job 110 entered on queue SYS$PRINT
  .
  .
  .

$ DELETE/ENTRY=110   SYS$PRINT
```

The PRINT command queues a copy of the file ALPHA.TXT in a HOLD status, to defer its printing until a later time. The system displays the job number, 110, and the name of the queue in which the file was entered. Later, the DELETE/ENTRY command requests that the entry be deleted from the queue SYS$PRINT.

2.
```
$ SUBMIT/HOLD/PARAMETERS=SCANLINE  DOFOR
  Job 203 entered on queue SYS$BATCH
$ SUBMIT/AFTER=18:00  WEATHER
  Job 210 entered on queue SYS$BATCH
   .
   .
   .

$ DELETE/ENTRY=(203,210)   SYS$BATCH
```

The SUBMIT commands spool 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 issue 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.

DELETE/SYMBOL

Deletes a symbol definition from a local symbol table or from the global symbol table, or deletes all symbol definitions in a symbol table. The /SYMBOL qualifier is required.

Format

```
    DELETE/SYMBOL   symbol-name


       Additional
    Command Qualifiers                 Defaults

    /ALL
    /GLOBAL                            /LOCAL
    /LOCAL                             /LOCAL
```

Prompts

Symbol: symbol-name

Command Parameters

symbol-name

Specifies the 1- through 255-character name of the symbol to be deleted. By default, the DELETE/SYMBOL command assumes the symbol is in the local symbol table for the current command procedure.

The symbol-name parameter is required unless /ALL is specified.

Description

If you specify both of the qualifiers /GLOBAL and /LOCAL, only the last one entered is accepted. The /SYMBOL qualifier must follow the DELETE command name.

For more information on how to use commands like this one in command procedures, consult the VAX/VMS Guide to Using Command Procedures.

Additional Command Qualifiers

/ALL

Specifies that all symbol names in the specified symbol table be deleted. If you do not specify either /LOCAL or /GLOBAL, all symbols defined at the current command level are deleted.

/GLOBAL

Indicates that the specified symbol name is in the global symbol table for the current process.

/LOCAL

Indicates that the symbol name is in the local symbol table for the current command level.

Examples

1.
```
$ DELETE/SYMBOL/ALL
```

   The DELETE/SYMBOL command deletes all symbol definitions at the current command level.

2.
```
$ DELETE/SYMBOL/GLOBAL PDEL
```

   The DELETE/SYMBOL command deletes the symbol named PDEL from the global symbol table for the process.

DEPOSIT

Replaces the contents of a specified location or locations in virtual memory.

The DEPOSIT command, together with the EXAMINE command, aids in debugging programs interactively. The DEPOSIT command is similar to the DEPOSIT command of the VAX-11 Symbolic Debugger.

You can truncate the DEPOSIT command to a single letter, D.

Format

```
    DEPOSIT    location=data[,...]

    Command Qualifiers          Defaults

    /ASCII                      None.
    /BYTE
    /DECIMAL
    /HEXADECIMAL
    /LONGWORD
    /OCTAL
    /WORD
```

Prompts

None.

Command Parameters

location

Specifies the starting virtual address of a location or series of locations whose contents are to be changed.

The specified location must be within the virtual address space of the image currently running in the process, and it must be accessible for both reading and writing for user access mode.

You can specify the location using any valid arithmetic expression. The expression can contain arithmetic or logical operators or symbol names which have been previously given values with DCL assignment statements. The DEPOSIT command automatically substitutes symbols with their current values when it evaluates the specified location.

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 using the symbol "." 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[,...]

Defines the data to be deposited into the specified location(s). If you specify a list, separate the items with commas; the DEPOSIT command writes the data in consecutive locations, beginning with the address specified.

By default, the data is assumed to be in hexadecimal format; the DEPOSIT command converts the data to binary format before writing it into the specified location. When non-ASCII data is deposited, the DEPOSIT command automatically performs symbol substitution when it evaluates data.

Description

When the DEPOSIT command completes, it displays the virtual memory address into which data is deposited and displays the new contents of the location, as follows:

```
address:  contents
```

If the address specified can be read, but not written, by the current access mode, the DEPOSIT command displays the original contents of the location. If the address specified 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, for example, 256. 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.

Restriction on Placement of Qualifiers: The DEPOSIT command analyzes expressions arithmetically. Therefore, qualifiers (which must be preceded by /) are interpreted correctly only when they appear immediately after the command name.

Command Qualifiers

/ASCII

Indicates that the specified data is ASCII. The DEPOSIT command converts the data to its binary equivalent before placing it in virtual memory.

When you specify ASCII data, the command interpreter compresses multiple blanks to a single blank before writing the data in memory; to deposit an ASCII string containing consecutive multiple blanks, enclose the string in quotation marks (").

When you specify /ASCII, or when ASCII mode is the default, any literal numeric values you enter are assumed to be hexadecimal.

/BYTE

Requests that data be deposited one byte at a time.

If you specify data values that are longer than a byte, an error occurs.

/DECIMAL

Indicates that the specified data is decimal; the DEPOSIT command converts the data to its binary equivalent before placing it in virtual memory.

/HEXADECIMAL

Indicates that the specified 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 specified 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 a word at a time.

Examples

1.
```
$ RUN MYPROG
    .
    .
    .

^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.

The CONTINUE command resumes execution of the image.

2.
```
$ DEPOSIT/ASCII   2C00=FILE: NAME: TYPE:
00002C00:  FILE: NAME: TYPE:...
```

The DEPOSIT command deposits character data at hexadecimal location 2C00 and displays the contents of the location after modifying it. Since the current default length is a longword, the response from the DEPOSIT command displays full longwords. Trailing dots 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 creates a listing of the records that do not match.

Format

```
DIFFERENCES   input-file-spec   [compare-file-spec]


Command Qualifiers                          Defaults

/COMMENT_DELIMITERS[=(characters[,...])]
/IGNORE=(characters[,...])
/LINE_WIDTH=n
/MATCH=size                                 /MATCH=3
/MAXIMUM_DIFFERENCES=n
/MERGED[=n]                                 /MERGED=1
/MODE=(radix[,...])                         /MODE=ASCII
/OUTPUT[=file-spec]                         (see text)
/SEPARATED
/SLP
/WINDOW=size                                /WINDOW=15


File Qualifiers                             Defaults

/CHANGE_BAR[=[c][,[NO]NUMBER]]              /CHANGE_BAR=-
                                               (!,NONUMBER)
```

Prompts

File 1: input-file-spec

File 2: compare-file-spec

Command Parameters

input-file-spec

Specifies the name of the primary input file to be compared.

The file specification must include a file name and file type. No wild card characters are allowed in the file specification.

compare-file-spec

Specifies the name of the secondary input file to be compared. Any nonspecified fields default to the corresponding field of the primary input file specification.

If you do not specify a secondary input file, the DIFFERENCES command uses the next lowest version of the primary input file.

No wild card characters are allowed in the file specification.

Description

Use the DIFFERENCES command to find out whether two files are identical and, if not, how they differ. DIFFERENCES compares the two files specified, on a record-by-record basis, and produces an output file that lists the differences, if any. Logical names are not fully supported by the DIFFERENCES command and should be avoided.

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 are:

  /COMMENT_DELIMITERS
  /IGNORE

  These qualifiers allow you to define characters that denote comments and characters to ignore while comparing files -- for example, extra blank lines, or tabs within lines.

  By default, DIFFERENCES compares every character in each record.

- Qualifiers that control the format of the information produced in the list of differences are:

  /CHANGE_BAR
  /LINE_WIDTH
  /MERGE
  /MODE
  /SEPARATED
  /SLP

  By default, DIFFERENCES merges differences it finds in the files being compared, and lists each record in the input file that has no match in the output file, followed by the first record that it finds that does have a match.

  You can specify combinations of qualifiers to request an output listing from DIFFERENCES that includes the comparison in more than one format. However, SLP output is incompatible with all other types of output.

- Qualifiers that control the extent of the comparison are:

  /MATCH
  /MAXIMUM_DIFFERENCES
  /WINDOW

  By default, DIFFERENCES reads every record in the primary input file, and looks for a matching record in the secondary input file. It terminates each search if it does not find a match within 15 records. Records are considered matched only if three sequential records are found in each file that are identical. Thus, a file containing less than three records always appears as not matched.

DIFFERENCES output is written by default to the current output device, that is, to SYS$OUTPUT. Use the /OUTPUT qualifier to request DIFFERENCES to write the output to an alternate file or device.

Command Qualifiers

/COMMENT_DELIMITERS[=(characters[,...])]

Requests that lines that are comments not be included in the comparison. If a specified comment character or characters appears as the first character in an input record, DIFFERENCES ignores the record in the comparison.

You can specify up to four comment characters. If you specify more than one, separate the characters with commas and enclose the list in parentheses.

You can specify characters either by using the character itself or by using one of the following keywords for special characters:

Keyword	Character
COLON	Colon (:)
COMMA	Comma (,)
EXCLAMATION	Exclamation point (!)
FORM	Form feed
LEFT	Left bracket ([)
RIGHT	Right bracket (])
SEMI	Semi-colon (;)
SLASH	Slash (/)
SPACE	Space
TAB	Horizontal tab

If you do not include a comment character, DIFFERENCES assumes the following default comment characters for the associated file types:

File Type	Default Comment Character
CBL	* and ;
CMD	! and ;
COM	!
FOR	! and C and D
All others	;

If the comment character (either explicitly or by default) is either an exclamation point (!) or semicolon (;), DIFFERENCES also ignores any comments on the right-hand side of a statement. If you also specify /IGNORE=TRAILING_BLANKS, DIFFERENCES ignores multiple blank spaces or tabs immediately preceding the comment character as well.

/IGNORE=(characters[,...])

Specifies one or more special characters to be ignored during the comparision. You can request DIFFERENCES to ignore the following:

BLANK_LINES	Blank lines between data lines
COMMENTS	Lines beginning with a comment character (use the /COMMENT_DELIMITERS qualifier to designate one or more non-default comment characters)
FORM_FEEDS	Form feed characters (DIFFERENCES removes form feed characters before comparing records)

TRAILING_BLANKS	Extra blank characters at the end of a line of text (DIFFERENCES strips trailing blanks and tabs before comparing records)
SPACING	Extra blank spaces or tabs within lines of text (DIFFERENCES compresses multiple blanks or tabs to a single blank before comparing records)

If you specify multiple types of characters to ignore, you must separate them by commas and enclose the list in parentheses.

By default, the DIFFERENCES command compares every character in each file and reports all differences.

/LINE_WIDTH=n

Specifies the width of lines in the output listing.

The minimum value for the line width, n, is 30.

By default, output is 132 characters wide, unless output is directed to the terminal. In this case, the output line width is controlled by the terminal line width.

/MATCH=size

Controls the number of records to be included in a match. The size value can be specified as a decimal number between 2 and 20.

By default, after DIFFERENCES finds unmatched records, it assumes that the files match after it finds 3 sequential records that match.

Specify a match size to override the default value of 3.

/MAXIMUM_DIFFERENCES=n

Specifies that the command is to terminate after n unmatching records have been found.

If you specify /MAXIMUM_DIFFERENCES, DIFFERENCES terminates after locating n unmatched records. The output file lists differences only on records compared until the maximum has been reached.

By default, DIFFERENCES compares every record in the specified input file.

/MERGED[=n]

Requests that the output file contain a merged list of differences. The value n is a decimal number from 1 to 10 indicating the number of matched records to list after each list of unmatched records.

By default, DIFFERENCES produces a merged listing, with one matched line listed after each set of unmatched records.

Use /MERGED 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 listing. You can request that the output be formatted in one or more radix modes by specifying the following keywords:

ASCII
HEXADECIMAL
OCTAL

You can truncate any of these keywords to 1 or more characters.

By default, DIFFERENCES writes the output file in ASCII. If you specify more than one code, the output listing contains the file comparison in each output mode.

When you specify more than one radix mode, separate them by commas and enclose the list in parentheses.

If you specify /SLP, the /MODE qualifier is ignored.

/OUTPUT[=file-spec]

Defines an output file to receive the output difference list. If you omit the /OUTPUT qualifier, the output is written to the current default output device (SYS$OUTPUT). If you specify /OUTPUT without a file specification, the output is directed to a file with the same file name as the input file and a file type of DIF. When you specify /OUTPUT, you can control the defaults applied to the output file specification, as described in Section 5.3.3, "Rules for Entering Output File Qualifiers." The output file type defaults to DIF.

No wild card characters are allowed in the file specification.

/SEPARATED

Creates a listing that contains a sequential list of unmatched records. Unmatched records in the primary input file are listed first, followed by unmatched records in the compare file.

By default, DIFFERENCES creates only a merged list of differences.

/SLP

Requests DIFFERENCES to produce an output file suitable for input to the SLP editor. If you specify /SLP you cannot specify any of the following output file qualifiers: /MERGED, /SEPARATED, /CHANGE_BAR.

Use the output file produced by the /SLP qualifier as input to SLP to update the primary input file specified, that is, to make the first input file match the second 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 primary input file and a filetype of DIF.

/WINDOW=size

Controls the number of records to search before listing a record as unmatched.

The window size is a decimal number with a minimum value of 5 and no maximum.

By default, the window size is 15, that is, DIFFERENCES searches 15 records in the compare file before listing a record as unmatched.

Specify a window size to control the number of records to search before listing the record as unmatched and continuing with the next record in the input file.

File Qualifiers

/CHANGE_BAR[=[c][,[NO]NUMBER]]

Requests that the output contain a listing of the associated file(s) with a change bar character next to the lines in the file that do not match.

The change bar character, c, specifies a one-character code that will appear in the left margin next to records that do not have a match.

By default, an exclamation point (!) is used as the change bar character.

You can also control whether the listing includes line numbers. You can specify:

NUMBER	Print line numbers
NONUMBER	Do not print line numbers

If not specified, NONUMBER is the default; that is, line numbers are not printed in the listing.

To specify both a change bar character and to request numbers, separate the options with a comma and enclose them in parentheses, for example, /CHANGE_BAR=(*,NUMBER).

Examples

Figure 1 shows the output from the DIFFERENCES command examples described below. Unless otherwise noted, the output is displayed on the current SYS$OUTPUT device.

1. `$ DIFFERENCES COPYDOC.COM`

   The DIFFERENCES command compares the contents of the two most recent versions of the file COPYDOC.COM in the current default directory. DIFFERENCES compares every character in every record and displays the results on the terminal.

2. `$ DIFFERENCES/IGNORE=(COMMENTS,SPACING) COPYDOC.COM`

   The DIFFERENCES command compares the same files as above, but ignores all comment lines (that is, all lines beginning with exclamation points), and ignores multiple blanks or tabs in input lines.

3.
   ```
   $ DIFFERENCES /OUTPUT -
   $_ COPYDOC.COM/SEPARATED/CHANGE_BAR=NUMBER
   DIF OUTPUT IN FILE SY:COPYDOC.DIF;1
   ```

   The DIFFERENCES command compares the same files as above, but requests two listings: one that lists the differences in each file separately, rather than merging them; and one that lists the first input file with change bar characters next to the lines that do not have a match in the second input file. The /CHANGE_BAR qualifier is a file qualifier; thus, only the file COPYDOC.COM;2 (the primary input file, by default) is listed with change bars indicating the lines that do not have an exact match in the file COPYDOC.COM;1.

   Both types of output are written to the default file, COPYDOC.DIF.

4. `$ DIFFERENCES COPYDOC.COM [MALCOLM.TESTFILES]`

   The DIFFERENCES command compares the highest existing versions of the file COPYDOC.COM in the current default directory with the copy in the directory MALCOLM.TESTFILES.

Output for Example 1

```
**************************************************************************
************************* FILE COMPARE UTILITY *************************
************************** DIF -- VERSION 1.12 *************************
**************************************************************************


**************************************************************************
********************** MERGED LIST OF DIFFERENCES **********************
**************************************************************************


FILE SY:COPYDOC.COM;3
    1  $ ! Command procedure to copy all files with file type of TXT
    2  $ ! into a master file named MASTER.DOC and to print 20 copies
    3  $ ! of MASTER.DOC
******************************
FILE SY:COPYDOC.COM;2
    1  $ ! Command Procedure to copy all files with file type of TXT
    2  $ ! into a master file named MASTER.DOC and to print 10 copies
    3  $ ! of MASTER.DOC
*******************************************************
*******************************************************
FILE SY:COPYDOC.COM;3
    6  $ DELETE MASTER.DOC;*
    7  $ PURGE *.TXT
    8  $ COPY *.TXT    MASTER.DOC
    9  $ PRINT/COPIES=20/LOWER MASTER.DOC
******************************
FILE SY:COPYDOC.COM;2
    6  $ PURGE *.TXT
    7  $ COPY *.TXT MASTER.DOC
    8  $ PRINT/COPIES=10/LOWER MASTER.DOC
```

Output for Example 2

```
**************************************************************************
************************* FILE COMPARE UTILITY *************************
************************** DIF -- VERSION 1.12 *************************
**************************************************************************


**************************************************************************
********************** MERGED LIST OF DIFFERENCES **********************
**************************************************************************


FILE SY:COPYDOC.COM;3
    6  $ DELETE MASTER.DOC;*
    7  $ PURGE *.TXT
    8  $ COPY *.TXT    MASTER.DOC
    9  $ PRINT/COPIES=20/LOWER MASTER.DOC
******************************
FILE SY:COPYDOC.COM;2
    6  $ PURGE *.TXT
    7  $ COPY *.TXT MASTER.DOC
    8  $ PRINT/COPIES=10/LOWER MASTER.DOC
```

Figure 1 Sample Output of DIFFERENCES Command

Output for Example 3

```
**************************************************************************
************************** FILE COMPARE UTILITY **************************
*************************** DIF -- VERSION 1.12 **************************
**************************************************************************


**************************************************************************
*********************** CHANGE-BAR OUTPUT FOR FILE ***********************
**************************** SY:COPYDOC.COM;3 ****************************
**************************************************************************


!       $ ! Command procedure to copy all files with file type of TXT
!       $ ! into a master file named MASTER.DOC and to print 20 copies
        $ ! of MASTER.DOC
        $ !
        $ !
!       $ DELETE MASTER.DOC;*
!       $ PURGE *.TXT
!       $ COPY *.TXT    MASTER.DOC
!       $ PRINT/COPIES=20/LOWER MASTER.DOC

**************************************************************************
*************************** LISTED DIFFERENCES ***************************
**************************************************************************


                    RECORDS FROM FILE SY:COPYDOC.COM;3
                  WITHOUT A MATCH IN FILE SY:COPYDOC.COM;2

    1   $ ! Command procedure to copy all files with file type of TXT
    2   $ ! into a master file named MASTER.DOC and to print 20 copies
    6   $ DELETE MASTER.DOC;*
    7   $ PURGE *.TXT
    8   $ COPY *.TXT    MASTER.DOC
    9   $ PRINT/COPIES=20/LOWER MASTER.DOC

                    RECORDS FROM FILE SY:COPYDOC.COM;2
                  WITHOUT A MATCH IN FILE SY:COPYDOC.COM;3

    1   $ ! Command Procedure to copy all files with file type of TXT
    2   $ ! into a master file named MASTER.DOC and to print 10 copies
    6   $ PURGE *.TXT
    7   $ COPY *.TXT MASTER.DOC
    8   $ PRINT/COPIES=10/LOWER MASTER.DOC
```

Output for Example 4

```
**************************************************************************
************************** FILE COMPARE UTILITY **************************
*************************** DIF -- VERSION 1.12 **************************
**************************************************************************


**************************************************************************
*********************** MERGED LIST OF DIFFERENCES ***********************
**************************************************************************


           NONE
```

Figure 1 (Cont.) Sample Output of DIFFERENCES Command

Provides a list of files or information about a file or group of files.

Format

```
DIRECTORY   [file-spec[,...]]

Command Qualifiers             Defaults

/BEFORE[=time]
/BRIEF                         /BRIEF
/COLUMNS=n                     /COLUMNS=4
/CREATED                       /CREATED
/[NO]DATE[=option]             /NODATE
/EXCLUDE=(file-spec[,...])
/EXPIRED                       /CREATED
/FULL                          /BRIEF
/[NO]HEADING                   /HEADING
/MODIFIED                      /CREATED
/OUTPUT[=file-spec]
/[NO]OWNER                     /NOOWNER
/PRINTER
/[NO]PROTECTION                /NOPROTECTION
/SINCE[=time]
/[NO]SIZE[=option]             /NOSIZE
/TOTAL                         /BRIEF
/[NO]TRAILING                  /TRAILING
/VERSIONS=n
```

Prompts

None.

Command Parameters

file-spec[,...]

Specifies one or more files to be listed. The syntax of a file specification determines what file(s) will be listed, as follows:

- If you do not enter a file specification, the DIRECTORY command lists all versions of the files in your 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 and file type, all versions of all files in the specified directory are listed.

- If a file specification contains a file name and/or file type and no version number, the DIRECTORY command lists all versions.
- If a file specification contains only a file name, the DIRECTORY command assumes all file types and versions.

If you specify more than one file, separate the file specifications with either commas (,) or plus signs (+). You can use wild card characters 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. See Section 2.1.6 for a full description of wild card characters.

Description

The output of the DIRECTORY command depends on certain formatting qualifiers and their defaults. These qualifiers are: /COLUMNS, /DATE, /FULL, /OWNER, /PROTECTION, and /SIZE. However, the files that are listed always appear in alphabetical order, with the highest-numbered versions first. The page width is adjusted automatically to the number of columns requested.

In studying the qualifiers and the capabilities they offer, watch for qualifiers that work together as well as qualifiers that override other qualifiers. For example, if you want the full format, you cannot expect that much information in more than one column, so if you specify both /COLUMNS and /FULL, the number of columns you request is ignored.

Command Qualifiers

/BEFORE[=time]

Specifies that only those files dated earlier than a particular time be printed. You can specify an absolute date and time. Observe the syntax rules for date and time values specified in Section 5.8.

This qualifier is normally used in conjunction with one of the following qualifiers: /CREATED, /EXPIRED, or /MODIFIED. If you omit the /BEFORE qualifier, you obtain all the files created, regardless of date. However, if you specify /BEFORE without a date or time, the default provides the files created prior to today.

/BRIEF

Includes only the file name, type, and version number of each file to be listed, as shown in Figure 2. The default output format is /BRIEF. However, the /BRIEF qualifier is overridden, whether specified explicitly or by default, whenever any of the following formatting qualifiers is specified in the command: /SIZE, /DATE, /OWNER, /PROTECTION, /NOHEADING, or /FULL.

The brief format lists the files in alphabetical order from left to right on each line, in descending version number order.

/COLUMNS=n

Lists the files using the specified number of columns on each line of the display. This qualifier is used in conjunction with the /BRIEF qualifier (either explicitly or by default). By default, the number of columns in the brief format is four; however, you may request the brief format with as many columns as you desire. When other formatting qualifiers are specified in the command, they override the /COLUMNS qualifier.

/CREATED

Selects the files according to their date of creation. This qualifier is relevant only when used with the /BEFORE or /SINCE qualifiers, and should not be used with the /EXPIRED or /MODIFIED qualifiers. By default, the selection of files according to some date and time always uses the creation date.

/DATE[=option]
/NODATE

Includes the creation, expiration, or date last written for each file listed. If you omit this qualifier, the default is /NODATE. However, if you specify /DATE without an option, the creation date is provided.

You may specify one of the following options with the /DATE qualifier:

ALL	Lists all three file dates in the order, left to right, CREATED, MODIFIED, EXPIRED.
CREATED	Lists the creation date with each file.
EXPIRED	Lists the expiration date with each file.
MODIFIED	Lists the last date the file was written.

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

Excludes the listed file specification(s) from the directory search. You may use wild card characters, as described in Section 2.1.6, for the file specification(s). At least one file specification is required for this qualifier, but the file specification must not include a device or directory specification. Separate multiple file specifications by commas, and enclose the list in parentheses.

/EXPIRED

Selects files according to the planned expiration date for each file. This qualifier is relevant only with the /BEFORE or /SINCE qualifiers, and should not be used with the /CREATED or /MODIFIED qualifiers.

/FULL

Lists the following items for each file:

- file name
- file type
- version number
- number of blocks used
- number of blocks allocated
- date of creation
- date last modified
- date of expiration
- file owner's UIC
- file protection
- file identification number (FID)
- file organization
- other file attributes
- record attributes
- record format

You can find descriptions of these items in the Introduction to VAX-11 Record Management Services.

The /FULL qualifier overrides the default brief listing format.

/HEADING
/NOHEADING

Controls whether heading lines consisting of a device description and directory specification are printed. The default output format provides this heading.

When you specify /NOHEADING, the output appears in single column format. In addition, the output contains the full file specification on every file. If you specify the /NOHEADING qualifier and also specify a value other than 1 with /COLUMNS, the number of columns you specify is disregarded.

You may find the combination of the /NOHEADING and /NOTRAILING qualifiers useful in command procedures where you want to create a list of complete file specifications for later operations.

/MODIFIED

Selects files according to the last date the file was modified. This qualifier is relevant only with the /BEFORE or /SINCE qualifiers, and should not be used with the /CREATED or /EXPIRED qualifiers.

/OUTPUT[=file-spec]

Requests that the DIRECTORY command output be written to the file specified rather than to the current SYS$OUTPUT device. If you specify the /OUTPUT qualifier without a file specification, the output is directed to SYS$OUTPUT. If you omit the file type in the file specification, the default file type is LIS.

No wild card characters are allowed in the file specification.

/OWNER
/NOOWNER

Controls whether the file's owner UIC is listed. By default, the owner UIC is not listed.

/PRINTER

Queues the command output for printing under the name given by the /OUTPUT qualifier. If you specify /PRINTER without the /OUTPUT qualifier, the output is directed to a file named DIRECTORY.LIS, which is spooled for printing automatically and then deleted.

/PROTECTION
/NOPROTECTION

Controls whether the file protection for each file is listed. The default is /NOPROTECTION, which does not list the file protection.

/SINCE[=time]

Specifies that only those files dated after a specified time be printed. You can specify an absolute date and time. Observe the syntax rules for date and time values specified in Section 5.8.

This qualifier is normally used in conjunction with one of the following qualifiers: /CREATED, /EXPIRED, or /MODIFIED. If you omit the /SINCE qualifier, you will obtain all the files created, regardless of date. However, if you specify /SINCE without a time or date, you will obtain all files created since today began.

/SIZE[=option]
/NOSIZE

Provides the file size in blocks used and/or allocated for each file listed, according to the option you specify. If you omit this qualifier, the default is /NOSIZE. However if you specify only /SIZE without an option, the listing provides the file size in blocks used, by default. The options you can specify are:

ALL	Lists both the file size in blocks used and allocated.
ALLOCATION	Lists the file size in blocks allocated.
USED	Lists the file size in blocks used.

/TOTAL

Inhibits the listing of all individual file information and prints only the trailing lines as described under the /TRAILING qualifier.

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 summarize the following information are output:

- number of files listed
- total number of blocks used per directory
- total number of blocks allocated
- total number of directories and total blocks used and/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. If you omit /SIZE or /FULL, only the number of files is printed and possibly the total number of directories, if applicable. If you specify /SIZE, the number of blocks is also printed, according to the size option selected (used and/or allocated). If you specify /FULL, the number of files and the number of blocks used and allocated are all given.

/VERSIONS=n

Causes the latest n versions of each of the files selected to be listed. If you omit the /VERSIONS qualifier, by default the listing includes all versions of each file.

Examples

1.
```
$ DIRECTORY
```

   The DIRECTORY command lists all versions of all files in the current default disk and directory in the brief format. The heading identifies the disk and directory, and the trailing line gives the total number of files.

2.
```
$ DIRECTORY/VERSIONS=1/COLUMNS=1 AVERAGE.*
```

   The DIRECTORY command lists only the highest versions of all files named AVERAGE in the current default directory. The format is brief, but restricted to just one column. Heading and trailing lines are provided.

3.
```
$ DIRECTORY BLOCK%%%
```

   The DIRECTORY command 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 output format is brief, in four columns, with heading and trailing lines.

4.
```
$ DIRECTORY/TOTAL/SIZE=ALL
```

The DIRECTORY command outputs only a header and a trailing line that identifies the total number of files and the blocks used and allocated for all versions of all files in the default disk and directory.

5.
```
$ DIRECTORY/EXCLUDE=(AVER.DAT;*,AVER.EXE;*) [*...]AVER
```

The DIRECTORY command locates all versions and types of files named AVER in all directories and subdirectories on the default disk. From this list all versions of all files named AVER.DAT and AVER.EXE are excluded prior to listing and totalling.

6.
```
$ DIRECTORY-
$_/MODIFIED/SINCE=09-JUL-1979:01:30/SIZE=ALL/OWNER-
$_/PROTECTION/OUTPUT=UPDATE/PRINTER [A*]
```

The DIRECTORY command locates all files that have been modified since 1:30 AM on July 9, 1979 and that reside on the default disk in directories whose names begin with the letter A. It formats the output to include all versions, the size used and allocated, the date last modified, the owner, and the protection codes. The output is directed to a file named UPDATE.LIS that is spooled automatically and deleted when done.

The following notes are keyed to the sample DIRECTORY command listings in Figure 2.

❶ Disk and directory name

❷ File name, file type, and version number of each file

❸ File identification number (FID) in the format:

(file-number,file-sequence-number,relative-volume-number)

❹ Number of blocks occupied by the file

❺ Number of blocks allocated for the file

❻ Date and time the file was created or last modified

❼ User identification code of the file's owner in the format:

[group,member]

❽ Protection code associated with the file, in the format:

[system,owner,group,world]

❾ Summary of file information, in the format:

Total of x files, in-use/allocated blocks.

❿ Date and time that this version of the file was last revised, and the revision number

⓫ Grand total of directory information in the format:

Grand total of x directories, y files.

```
$ DIRECTORY AVERAGE.*

Directory DBA1:[MALCOLM]  ❶

AVERAGE.EXE;9 ❷   AVERAGE.FOR;6      AVERAGE.LIS;4     AVERAGE.OBJ;12

Total of 4 files. ❾


$ DIRECTORY/SIZE=USED/DATE=CREATED/VERSIONS=1/PROTECTION  AVERAGE

Directory DBA1:[MALCOLM] ❶

AVERAGE.EXE;9 ❷      6 ❹   ❻ 10-JUL-1979 15:43 (RWED,RWED,RWED,RE) ❽
AVERAGE.FOR;6        2         2-JUL-1979 10:29 (RWED,RWED,RWED,RE)
AVERAGE.LIS;4        5         9-JUL-1979 16:27 (RWED,RWED,RWED,RE)
AVERAGE.OBJ;12       2         9-JUL-1979 16:27 (RWED,RWED,RWED,RE)

Total of 4 files, 15 blocks. ❾


$ DIRECTORY/FULL/VERSIONS=1   [MALCOLM.*]AVERAGE

Directory DBA1:[MALCOLM.AAA] ❶

AVERAGE.EXE;9 ❷      Size:      ❹ 6/6 ❺      Created: 10-JUL-1979 15:43 ❻
                     Owner:     [360,007] ❼   Revised: 10-JUL-1979 15:43(9)❿
                     File ID: (3680,40,0) ❸   Expires: <None specified>
  File protection:   System:RWED, Owner:RWED, Group:RWED, World:RE ❽
  File organization: Sequential
  File attributes:   Allocation=6, Extend=0, Contiguous-best-try
  Record format:     Fixed length 512 byte records
  Record attributes: None

                     .
                     .
                     .

Total of 4 files, 23/30 blocks. ❾
                     .
                     .
                     .

Grand total of 4 directories, 10 files.  ⓫
```

Figure 2 Sample Output of DIRECTORY Command

DISMOUNT

Releases a volume previously accessed with a MOUNT command.

Format

```
DISMOUNT   device-name[:]

Command Qualifiers       Defaults

/UNIT
/[NO]UNLOAD              /UNLOAD
```

Prompts

Device: device-name[:]

Command Parameters

device-name[:]

Specifies the name of the device to be dismounted. You can specify a physical device name or a logical name assigned to a physical device name. If you omit a controller designation and/or a unit number, they default to controller A, and unit 0, respectively.

If the volume that is currently mounted on the device is a member of a disk or tape volume set, all volumes in the set are dismounted, unless /UNIT is specified.

Description

If the volume was mounted with the /SHARE qualifier, it is not actually dismounted until all users who mounted it dismount it. 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 /GROUP or /SYSTEM, 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 dismounting of a volume is done by the file system and is not completed until all open files on the volume have been closed. Thus, a substantial amount of time can pass between the time you issue the DISMOUNT command and the completion of the dismount. Always wait for the drive to unload before you remove the volume. (You can verify that the dismount has completed by issuing the SHOW DEVICES command.)

Command Qualifiers

/UNIT

Specifies, for disk volume sets, that only the volume on the specified device is dismounted. By default, the DISMOUNT command dismounts all volumes in a volume set.

/UNLOAD
/NOUNLOAD

Controls whether the DISMOUNT command unloads the physical device on which the volume is mounted and makes the device not ready.

By default, the DISMOUNT command unloads the device. Use the /NOUNLOAD qualifier to keep the device and volume in a ready state.

Examples

1.
```
$ MOUNT MT:  PAYVOL TAPE
   .
   .
   .
$ DISMOUNT TAPE
```

The MOUNT command 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 mounts the volume labeled DOC_FILES on the device DBA3. Other users can issue 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.

3.
```
$ DIRECTORY DMA2:[*]
No files found.
$ DISMOUNT/NOUNLOAD DMA2:
$ INITIALIZE DMA2:  BACK_UP
```

The DIRECTORY command ensures that no files remain on the RK06/RK07 volume mounted on the device DMA2. The DISMOUNT dismounts the volume; the /NOUNLOAD qualifier requests that the volume remain in a ready state. Then, the INITIALIZE command reinitializes the volume.

DUMP

Displays or prints the contents of a file or volume in ASCII, decimal, hexadecimal, or octal data format.

Format

```
DUMP   file-spec
```

Command Qualifiers	Defaults
/ASCII	/HEXADECIMAL
/BLOCKS=(START:n,END:m)	
/BYTE	/WORD
/DECIMAL	/HEXADECIMAL
/FILE_HEADER	
/[NO]FORMATTED	/FORMATTED
/HEADER	
/HEXADECIMAL	/HEXADECIMAL
/LONGWORD	/WORD
/NUMBER[=n]	
/OCTAL	/HEXADECIMAL
/OUTPUT[=file-spec]	/OUTPUT=SYS$OUTPUT
/PRINTER	
/RECORDS	
/WORD	/WORD

Prompts

File: file-spec

Command Parameters

file-spec

Specifies the file or volume whose contents are to be displayed.

No wild card characters are allowed in the file specification.

Description

By default, the DUMP command formats its output in hexadecimal words. You can specify the precise format of the dump by including a radix qualifier (/ASCII, /OCTAL, /DECIMAL, or /HEXADECIMAL) and/or a length qualifier (/BYTE, /WORD, or /LONGWORD). The valid combinations of these qualifiers are:

```
/BYTE/HEXADECIMAL
/BYTE/OCTAL
/WORD/DECIMAL
/WORD/HEXADECIMAL
/WORD/OCTAL
/LONGWORD/HEXADECIMAL
```

All other combinations are invalid.

Dumping Files: When you dump files, you can request that the entire file be dumped, or you can specify the /BLOCKS qualifier to indicate a range of virtually contiguous blocks in the file to be dumped.

The volume that contains the file must be mounted.

Dumping Volumes: When you dump volumes, the /BLOCKS qualifier is required; use it to specify a range of logically contiguous blocks to be dumped. (On a disk volume, blocks are 512 bytes long. On a tape volume, each record is an individual block; the maximum block length that DUMP can handle is 2048 bytes.)

The volume to be dumped must be allocated and mounted with the /FOREIGN qualifier of the MOUNT command. You must have the user privilege LOG_IO to dump the contents of a volume.

Reading Dumps: Hexadecimal dumps are read right to left, while octal dumps are read left to right.

Command Qualifiers

/ASCII

Requests that the dump be interpreted as ASCII data. When you specify /ASCII, the DUMP command prints control characters with a circumflex or up-arrow (^) preceding them, and it prints lowercase letters with a percent sign (%) preceding their uppercase equivalents.

If you specify /ASCII, any other radix and length qualifiers are invalid.

/BLOCKS=(START:n,END:m)

Specifies a range of blocks to be dumped, where n is the starting logical block number and m is the ending block number.

By default, the DUMP command prints the entire contents of a file. The /BLOCKS qualifier is required when you are dumping the contents of a volume.

/BYTE

Requests that the output file be formatted in bytes.

If you specify /BYTE, you cannot specify /DECIMAL.

/DECIMAL

Requests that the dump be printed in decimal format.

If you specify /DECIMAL, you cannot specify /LONGWORD or /BYTE.

/FILE_HEADER

Dumps each data block that has a Files-11 header structure in Files-11 header format. All other data blocks are output as dictated by the other qualifiers.

/FORMATTED
/NOFORMATTED

Controls whether headers are dumped in a formatted or unformatted display. This qualifier is meaningful only in conjunction with the /HEADER qualifier. If you specify /FORMATTED, the header is dumped in Files-11 format. If you specify /NOFORMATTED, the header is dumped in octal format.

/HEADER

Requests that the dump include the file header. To dump only the file header, specify /BLOCKS=(END:0).

You may control the header format display through the /FORMATTED qualifier.

If you omit the /HEADER qualifier, file headers are not dumped. If you specify /HEADER without the /FORMATTED qualifier, by default the file headers are formatted.

/HEXADECIMAL

Requests that the dump be printed in hexadecimal format.

/LONGWORD

Requests that the dump be formatted in longwords.

If you specify /LONGWORD, you cannot specify /OCTAL, /DECIMAL, or /ASCII.

/NUMBER[=n]

Controls line numbers assigned to records as they are dumped. If you specify /NUMBER, records in each block are numbered beginning with 0. If you specify a value for n, that number is assigned to the first line in the file.

By default, DUMP numbers all lines in the file consecutively, and does not number lines according to the blocks they are in.

/OCTAL

Requests that the dump be printed in octal format.

If you specify /OCTAL, you cannot specify /ASCII or /LONGWORD.

/OUTPUT[=file-spec]

Requests that the output listing from the DUMP command be written to the specified file or device. By default, the DUMP command displays the output on the current SYS$OUTPUT device. If you specify /OUTPUT and do not include a file specification, DUMP writes the output to a file with the same file name as the input file and a file type of DMP.

No wild card characters are allowed in the file specification.

/PRINTER

Requests that output be queued to the system printer. By default, DUMP writes to the current SYS$OUTPUT device. If you specify /PRINTER, the DUMP command names the print job FILDMP.DMP. If you specify /PRINTER, you cannot specify /OUTPUT.

/RECORDS

Requests that the dump be printed a record at a time, rather than a block at a time.

/WORD

Requests that the dump be formatted in words.

If you specify /WORD, you cannot specify /ASCII.

Examples

1. $ DUMP ORION.EXE

```
Dump of DB1:[122001.SSTEST]ORION.EXE;17 - File ID 1637,5,0
                    Virtual block 0,000001 - size 512. bytes

3130 3230 0000 0000 0044 0038 0028 007C          0000
0000 0000 0000 0000 0000 0000 0000 0101          0010
          .          .          .                  .
          .          .          .                  .
          .          .          .                  .
```

The DUMP command displays the contents of the image ORION.EXE in hexadecimal format, beginning with the first block in the file.

2. $ DUMP/ASCII/RECORDS ALPHA.TXT

```
Dump of DB1:[122001.CLUG]ALPHA.TXT;1 - File ID 5767.60,0
                    Record number 00. - size 5. bytes

000000    %A %A %A %A %A ^@ ^@ ^@ ^@ ^@ ^@ ^@ ^@ ^@ ^@ ^@

                    Record number 01. - size 5. bytes

000000    %B %B %B %B %B ^@ ^@ ^@ ^@ ^@ ^@ ^@ ^@ ^@ ^@ ^@
   .               .          .          .
   .               .          .          .
   .               .          .          .
```

The DUMP command displays the contents of the file ALPHA.TXT, a record at a time.

EDIT/EDT

EDIT/EDT

Invokes the EDT screen-oriented editor. The EDT editor is described in detail in the VAX-11 EDT Editor Reference Manual.

The /EDT qualifier is required.

Format

```
EDIT/EDT file-spec


   Additional
Command Qualifiers              Defaults

/[NO]COMMAND[=file-spec]        /COMMAND=EDTINI.EDT
/[NO]JOURNAL[=file-spec]        (see text)
/[NO]OUTPUT[=file-spec]         (see text)
/[NO]READ_ONLY                  /NOREAD_ONLY
/[NO]RECOVER                    /NORECOVER
```

Prompts

File: file-spec

Command Parameters

file-spec

Specifies the file to be created or edited using the EDT editor. If the file does not exist, it is created.

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 wild card characters are allowed in the file specification.

Description

The EDT editor creates or edits files. You can use EDT to enter or edit text in screen mode or line mode (or both). EDT begins in line mode. If you are editing an existing file, EDT prints the line number and text for the first line of the file. If you are creating a new file, EDT prints the following message:

```
Input file not found
[EOB]
```

In either case, EDT prints its prompt, which is the asterisk (*).

For complete details on the EDT editor, see the VAX-11 EDT Editor Reference Manual.

Additional Command Qualifiers

/COMMAND[=file-spec]
/NOCOMMAND

Controls whether EDT reads an initial command file before prompting at the terminal. If you specify a command file, EDT executes all commands in the file before beginning the editing session at the terminal. If you specify the /NOCOMMAND qualifier, EDT reads no command file before beginning the terminal session.

By default, an attempt is made to read from the default command file, EDITINI.EDT. If this default file does not exist, the editing session begins, without an error message. However, if you specify a command file that does not exist, EDT issues an error message and terminates the session.

No wild card characters are allowed in the file specification.

/JOURNAL[=file-spec]
/NOJOURNAL

Controls whether a journal file is created for the editing session. If you specify a journal file, this file contains all EDT commands you enter until an exit occurs. If your editing session ends abnormally (perhaps when you type CTRL/Y), you can invoke EDT again, and (using the /RECOVER qualifier) reinstate all commands from the aborted session. If you specify the /NOJOURNAL qualifier, no journal file is generated.

If you omit the /JOURNAL qualifier, or if you specify the qualifier without a file specification, the editor creates a journal file with the same file name as your input file and a default file type of JOU.

No wild card characters are allowed in the file specification.

/OUTPUT=file-spec
/NOOUTPUT

Defines the file specification of the file created during the editing session. If you do not specify the /OUTPUT qualifier, the output file has the same file name and type as the input file, and a version number one higher than the highest existing version of the file.

No wild card characters are allowed in the file specification.

You can suppress the creation of the output file by specifying /NOOUTPUT.

/READ_ONLY
/NOREAD_ONLY

Controls whether journaling and the creation of an output file are disabled. The /READ_ONLY qualifier is equivalent to specifying /NOOUTPUT and /NOJOURNAL. You might use the /READ_ONLY qualifier to edit files where you are denied write access.

The default is /NOREAD_ONLY, which has no effect on journaling or output.

/RECOVER
/NORECOVER

Determines whether or not EDT reads commands from a journal file prior to starting the editing session. The default is /NORECOVER, which omits the step.

The /RECOVER qualifier requests EDT to open the input file and then read EDT commands from the file specified by inputfilename.JOU (which was created by EDT's /JOURNAL feature). This restores all commands that were lost in a previously aborted editing session.

The /RECOVER qualifier does not accept a file-spec; therefore, if the recovery file has a name different from inputfilename.JOU, you must specify both the /JOURNAL qualifier and the /RECOVER qualifier to obtain it for recovery.

Examples

1.
```
$ EDIT/EDT OLDFILE.TXT/OUTPUT=NEWFILE.TXT
   1          (first line of file text)
*
```

This EDIT/EDT command invokes the EDT editor for editing the file OLDFILE.TXT. EDT reads commands from EDTINI.EDT, if that file exists; then the terminal editing session begins. When the session ends, the edited file has the name NEWFILE.TXT.

2.
```
$ EDIT/EDT OLDFILE.TXT/RECOVER
```

This EDIT/EDT command invokes the EDT editor for recovering from an abnormal exit during a previous session. EDT opens the file OLDFILE.TXT, and then reads all editing commands from the previous session (from the file OLDTEXT.JOU). Once the old commands have been processed, the editing session continues at the terminal.

Invokes the VAX/VMS SLP editor, which is described in detail in the VAX-11 Utilities Reference Manual.

The /SLP qualifier is required.

Format

```
EDIT/SLP   file-spec


    Additional
Command Qualifiers                       Defaults

/[NO]AUDIT_TRAIL[=(option[,...])]        /AUDIT_TRAIL=(POSITION:80 -
                                            SIZE:8)
/[NO]CHECKSUM[=value]                    /NOCHECKSUM
/LIST[=file-spec]
/[NO]OUTPUT[=file-spec]                  (see text)
/[NO]REPORT                              /NOREPORT
/[NO]TAB                                 /NOTAB
/[NO]TRUNCATE[=position]                 /NOTRUNCATE
```

Prompts

File: file-spec

Command Parameters

file-spec

Specifies the file to be edited. If you do not include a file type, it is null by default.

The file should be a disk file on a Files-11 formatted volume.

No wild card characters are allowed in the file specification.

Additional Command Qualifiers

These qualifiers can be overridden in the SLP input file. For complete details on any of these qualifiers, see the VAX-11 Utilities Reference Manual.

/AUDIT_TRAIL[=(option[,...])]
/NOAUDIT_TRAIL

Controls whether records in the output file from SLP contain an audit trail, and optionally defines the location of the audit trail. You can specify one or both of the following options:

POSITION:n	Define the starting character position of the audit trail; by default, the audit trail is placed in column 80. If you specify this option, SLP rounds the value, n, to the next highest tab stop.
SIZE:n	Define the number of characters in the audit trail; by default, the audit trail is 8 characters.

If you specify more than one option, separate them by commas and enclose the list in parentheses.

If you specify /NOAUDIT_TRAIL, the output file does not contain a record of changes.

/CHECKSUM[=value]
/NOCHECKSUM

Controls whether a checksum is calculated for the edit commands. If you specify the /CHECKSUM qualifier without a value, SLP calculates and reports the checksum on your terminal. If you specify a value that differs from the one SLP calculates, SLP displays a warning message but completes the edit.

The default is /NOCHECKSUM, which does not calculate a checksum.

/LIST[=file-spec]

Creates a line-numbered listing of a file. By default, no line-numbered listing is produced. Use /LIST when you want a listing of lines in sequential order. If you do not specify a file specification with /LIST, SLP uses the same file name as the input file and a file type of LST. If you enter a file specification that does not include a file type, SLP uses the default file type of LST.

No wild card characters are allowed in the file specification.

/OUTPUT=file-spec
/NOOUTPUT

Defines the file specification of the output file created during the editing session, if any. If you do not specify /OUTPUT, the output file has the same file name and type as the input file, and a version number one higher than the highest existing version of the file.

No wild card characters are allowed in the file specification.

You can suppress the creation of the output file by specifying /NOOUTPUT.

/REPORT
/NOREPORT

Controls whether line truncations that result from audit trails are reported. If you specify the /REPORT qualifier, not only will warning messages appear on the terminal, but the listing file will contain a question mark (?) in place of the period (.) in the line number of all truncated lines.

The default is /NOREPORT, which does not report line truncations.

/TAB
/NOTAB

Controls whether SLP places spaces or tabs at the end of each record containing an audit trail. The default is /NOTAB which causes SLP to insert spaces at the end of each record that contains an audit trail.

If you specify the /TAB qualifier, SLP inserts tabs at the end of each record that contains an audit trail.

/TRUNCATE[=position]
/NOTRUNCATE

Requests SLP to truncate each record in the input file at a specified column when it creates the output file. This qualifier allows you to delete an audit trail from a file previously updated with SLP. The default is not to truncate the records.

If you do not specify a position with the /TRUNCATE qualifier, SLP truncates input records at the beginning position of the audit trail.

Examples

1.

```
$ EDIT/SLP  AVERAGE.FOR
    .
    .
    .

/
$
```

The command procedure illustrated uses the EDIT/SLP command; all input lines for the SLP editor follow the command in the input stream, and are terminated by the slash character (/).

2.

```
$ EDIT/SLP/LIST=AVLST  AVERAGE.FOR
    .
    .
    .

/
SLP>
^Z
$
```

This interactive editing session with the SLP editor requests a line-numbered listing, AVLST.LST, as output. The CTRL/Z terminates the session.

EDIT/SOS

Invokes the VAX/VMS SOS editor, which is described in detail in the VAX-11 Text Editing Reference Manual.

The /SOS qualifier is not required; SOS is the VAX/VMS default editor.

Format

```
    EDIT/SOS   file-spec


        Additional
    Command Qualifiers                   Defaults

    /[NO]BAK                             /BAK
    /[NO]DECIDE                          /NODECIDE
    /[NO]EXACT                           /NOEXACT
    /[NO]EXPERT                          /NOEXPERT
    /INCREMENT=n                         /INCREMENT=100
    /ISAVE=n                             /ISAVE=0
    /[NO]LINE                            /LINE
    /[NO]LOWER                           /LOWER
    /[NO]NUMBERS                         /NUMBERS
    /[NO]OUTPUT[=file-spec]              (see text)
    /PLINES=n                            /PLINES=16
    /[NO]READ_ONLY                       /NOREAD_ONLY
    /SAVE=n                              /SAVE=0
    /START=n                             /START=100
    /STEP=n                              /STEP=100
```

Prompts

File: file-spec

Command Parameters

file-spec

Specifies the file to be created or edited. If you do not include a file type, it is null by default.

The file, if it exists, must be a disk file on a Files-11 formatted volume. If the file you specify does not exist, it is created.

No wild card characters are allowed in the file specification.

Additional Command Qualifiers

The settings defined by some of these qualifiers can be overridden during the SOS session. For complete details on these qualifiers, see the VAX-11 Text Editing Reference Manual.

/BAK
/NOBAK

Controls whether SOS increments the version number of the output file when you issue the first SOS Save World command in the session or you issue an SOS End command without a previous Save World command.

By default, SOS increments the version number. If you want to overwrite the current version, specify /NOBAK.

/DECIDE
/NODECIDE

Controls whether SOS automatically enters Decide mode following each Substitute command. The default is not to enter Decide Mode.

/EXACT
/NOEXACT

Controls whether SOS matches character strings in Find and Substitute commands exactly or treats uppercase and lowercase letters as equivalent. The default is to treat uppercase and lowercase letters as equivalent.

/EXPERT
/NOEXPERT

Controls whether SOS displays the long form of error messages, requests confirmation of deletions, or displays various informational messages during the terminal session. The default is /NOEXPERT, which provides all this assistance.

/INCREMENT=n

Specifies the line number increment you want SOS to use as the default when you insert new lines in the file. If /INCREMENT is not specified, the line number increment is 100 by default.

/ISAVE=n

Requests SOS to issue a Save World command automatically after every n new lines of text that you insert with the Insert or Replace commands. Unless /ISAVE is specified, no saving of new input lines occurs.

/LINE
/NOLINE

Indicates whether SOS should use the existing line numbers when you edit a file, or should renumber the lines when it opens the file for editing. By default, SOS uses the current line numbers in the file.

/LOWER
/NOLOWER

Indicates whether SOS should accept all lowercase letters as they are entered or should translate all lowercase letters to uppercase. By default, SOS accepts lowercase letters. The /LOWER qualifier has no effect on data that already exists in a file.

/NUMBERS
/NONUMBERS

Controls whether SOS prints out the line numbers that may be present in an input file. By default, SOS prints the line numbers. Specify /NONUMBERS if you want to suppress the display of the line numbers.

/OUTPUT=file-spec
/NOOUTPUT

Defines the file specification of the output file created during the editing session, if any. If you do not specify /OUTPUT, the output file has the same file name and type as the input file, and a version number one higher than the highest existing version of the file.

You can suppress the creation of the output file by specifying /NOOUTPUT.

/PLINES=n

Specifies the number of lines that SOS prints each time you issue the SOS Print command. If /PLINES is not specified, 16 lines are printed by default.

/READ_ONLY
/NOREAD_ONLY

Controls whether SOS opens the input file for reading and writing or only for reading. By default, SOS opens files for reading and writing.

/SAVE=n

Requests SOS to automatically issue a Save World command after every n SOS commands that change text. If /SAVE is not specified, no saving of changes occurs.

/START=n

Specifies the line number you want to assign to the first line in the file and to each new page in the file. This value also controls the line number increment SOS uses when you issue the reNumber command. If /START is not specified, line numbering starts with 100 by default.

/STEP=n

Specifies the line number increment for SOS to use when it assigns line numbers to existing files that do not have line numbers. If /STEP is not specified, the line number increment is 100 by default.

Examples

1.
```
$ EDIT/SOS/OUTPUT=TEST.FOR  ACCOUNT.FOR/PLINES=10
```

   The EDIT/SOS command invokes the SOS editor, to edit the file ACCOUNT.FOR. The /PLINES qualifier sets the default number of lines to print with each SOS Print command during the editing session. When the edit session is terminated, the changes are written into the file TEST.FOR.

Invokes the SUMSLP batch-oriented editor, to update a single input file with multiple files of edit commands.

The SUMSLP editor is described in detail in the VAX-11 Utilities Reference Manual.

The /SUM qualifier is required.

Format

```
EDIT/SUM   file-spec

    Additional
Command Qualifiers                     Defaults

/LIST[=file-spec]
/[NO]OUTPUT[=file-spec]                (see text)

File Qualifiers                        Defaults

/UPDATE[=(update-file-spec[,...])]     None.
```

Prompts

File: file-spec

Command Parameters

file-spec

Specifies the file to be edited. If you do not specify a file type, it is null by default.

The file should be a disk file on a Files-11 formatted volume.

No wild card characters are allowed in the file specification.

Description

The SUMSLP editor supplements the functions of SLP by allowing multiple command files to be applied to a single input file. However, certain fixed rules direct how the command files are combined for the update. Read the VAX-11 Utilities Reference Manual description of these rules.

Additional Command Qualifiers

/LIST[=file-spec]

Requests a line-numbered listing file. The listing file shows the original lines, the inserted lines, and an audit trail. By default there is no listing file. If you specify /LIST without a file specification, by default the listing file has the same file name as the input file. The default file type is LIS.

No wild card characters are allowed in the file specification.

/OUTPUT[=file-spec]
/NOOUTPUT

Controls whether an output file is created for the editing session. If you do not specify /OUTPUT, by default an output file is created with the same file name and file type as the input file, and a version number one higher than the highest existing file version number.

You can suppress the creation of the output file by specifying /NOOUTPUT. This could be useful if all you need is a line-numbered listing file.

No wild card characters are allowed in the file specification.

File Qualifiers

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

Provides the file specification of one or more files containing the editing commands and changes to be applied to the input source file.

All update files must be disk files on a Files-11 formatted volume.

If you omit the /UPDATE file specification, by default SUMSLP updates from a file with the same name as the input file and a file type of UPD. If you omit the /UPDATE qualifier entirely, no updating occurs; however, if you specify only /LIST, you obtain a numbered listing.

If you specify multiple update files, separate them by commas and enclose the list in parentheses. Note that if you omit fields in the file specifications in the list, the default value is taken from the immediately preceding file specification. When you specify multiple updates files, the files are combined according to rules described in the VAX-11 Utilities Reference Manual.

No wild card characters are allowed in the file specification.

Examples

1.
```
$ EDIT/SUM/LIST=FILE1.LST FILE1.MAR/UPDATE
```

   The input source file FILE1.MAR is updated from the command file FILE1.UPD, creating a line-numbered listing file, FILE1.LST. A higher numbered file named FILE1.MAR contains the results.

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. This command is required only if the DECK command preceded input data in the command stream, or 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.

For more information on how to use commands like this one in command procedures, consult the VAX/VMS Guide to Using Command Procedures.

Format

```
$ EOD
```

Command Qualifiers	Defaults
None.	None.

Prompts

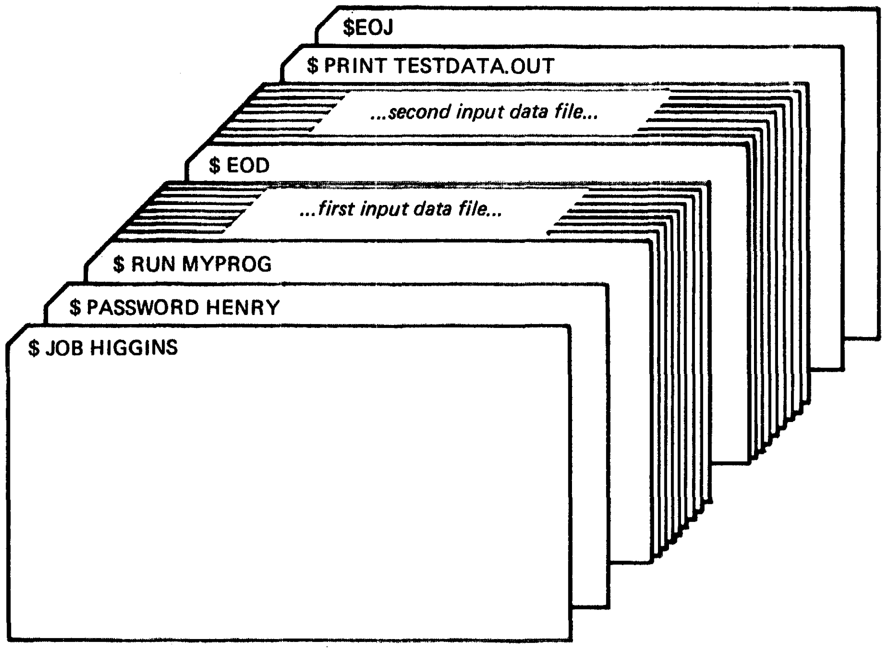
None.

Command Parameters

None.

Examples

1.



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.

For additional examples, see the discussion of the DECK command.

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 ($).

For more information on how to use commands like this one in command procedures, consult the VAX/VMS Guide to Using Command Procedures.

Format

```
$ EOJ
```

Command Qualifiers	Defaults
None.	None.

Prompts

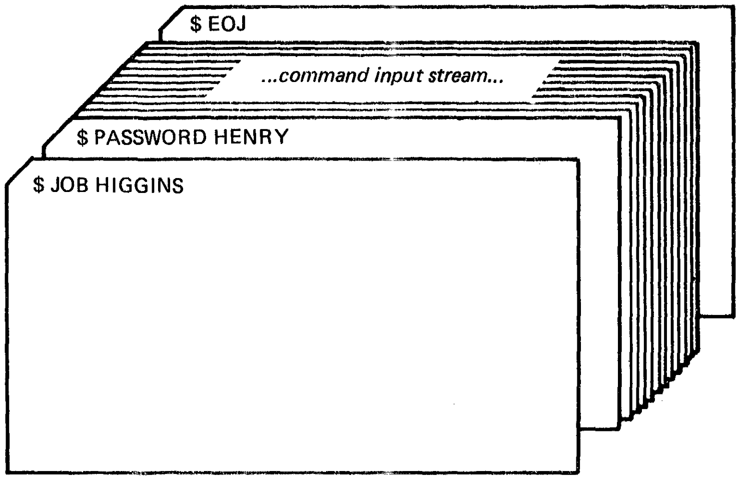
None.

Command Parameters

None.

Examples

1.



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.

You can truncate the EXAMINE command to a single letter, E.

Format

```
EXAMINE   location[:location]
```

Command Qualifiers	Defaults
/ASCII	None.
/BYTE	
/DECIMAL	
/HEXADECIMAL	
/LONGWORD	
/OCTAL	
/WORD	

Prompts

None.

Command Parameters

location[:location]

Specifies a virtual address or a range of virtual addresses whose contents you want to examine. If you specify a range of addresses, separate them with a colon (:); the second address must be larger than the first.

You can specify locations using any valid arithmetic expression that contains arithmetic or logical operators or symbol names that have been previously given values with DCL assignment statements.

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 symbol "." 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, asterisks (****) are displayed in the contents field.

Radix Qualifiers: The radix default for a DEPOSIT or EXAMINE command determines how the commands interpret numeric literals, for example 256. 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 (which must be preceded by /) are interpreted correctly only when they appear immediately after the command name.

Command 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 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
^Y
$ EXAMINE   2678
0002678:  1F4C5026
$ CONTINUE
```

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 hexadecimal 2678.

2.
```
$ BASE = %X1C00
$ READBUF = BASE + %X50
$ ENDBUF = BASE + %XA0
$ RUN   TEST
^Y
$ EXAMINE/ASCII READBUF:ENDBUF
00001C50:  BEGINNING OF FILE MAPPED TO GLOBAL SECTION
     .
     .
     .
```

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.

Terminates processing of the current command procedure. If the command procedure was executed from within another command procedure, control returns to the calling procedure.

If a command procedure is not being executed, the EXIT command terminates the current image.

Format

```
EXIT   [status-code]

Command Qualifiers                 Defaults

None.                              None.
```

Prompts

None.

Command Parameters

status-code

Defines 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 do not specify a status code, the current value of $STATUS is not changed and control returns to the outer level with the status of the most recently executed command or program.

Description

When a command procedure returns control to the interactive command level, the command interpreter uses the current value of $STATUS to locate and display the system message associated with the value, if any. Note that even numeric values generally produce warning, error, and fatal error messages and that odd numeric values generally produce either no message or a success or informational message. Example 4, below, shows how to use the EXIT command to determine the message and symbolic error name associated with a hexadecimal status code.

For more information on how to use commands like this one in command procedures, consult the VAX/VMS Guide to Using Command Procedures.

The EXIT command, when used in conjunction 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. For this reason, the EXIT command is generally preferable to the STOP command.

Examples

1.
```
$ ON WARNING THEN EXIT
$ FORTRAN 'P1'
$ LINK 'P1'
$ RUN 'P1'
```

The EXIT command 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.

2.
```
$ @SUBTEST
$ IF $STATUS .EQ. 7 THEN GOTO PROCESS
    .
    .
    .
$ EXIT
$ PROCESS:
    .
    .
    .
```

This procedure executes a second procedure, named SUBTEST.COM. When SUBTEST.COM completes, the outer procedure tests the value of the symbol $STATUS which may be set by SUBTEST as follows:

```
$ PATH1:
    .
    .
    .
$ EXIT 7
$ PATH2:
    .
    .
    .
$ EXIT 9
```

3.
```
$ 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
```

A command procedure tests whether a parameter was passed to it; if not, it 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.

4.
```
$ IF P1.EQS."" THEN INQUIRE P1 "Code"
$ CODE = %X'P1'
$ EXIT 'CODE'
```

This short command procedure 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.

5.
```
$ RUN MYPROG
^Y
$ EXIT
```

The RUN command initiates execution of the image MYPROG.EXE. Then the CTRL/Y interrupts the execution. The EXIT command that follows calls any exit handlers declared by the image before terminating MYPROG.EXE.

FORTRAN

Invokes the VAX-11 FORTRAN[1] compiler to compile one or more source programs. This command is described in detail in the VAX-11 FORTRAN User's Guide.

Format

```
FORTRAN   file-spec[,...]

Command Qualifiers               Defaults

/[NO]CHECK[=(option[,...])]      /CHECK=(NOBOUNDS,OVERFLOW)
/CONTINUATIONS=n                 /CONTINUATIONS=19
/[NO]DEBUG[=(option[,...])]      /DEBUG=(NOSYMBOLS,TRACEBACK)
/[NO]D_LINES                     /NOD_LINES
/[NO]F77                         /F77
/[NO]G_FLOATING                  /NOG_FLOATING
/[NO]I4                          /I4
/[NO]LIST[=file-spec]            (see text)
/[NO]MACHINE_CODE                /NOMACHINE_CODE
/[NO]OBJECT[=file-spec]          (see text)
/[NO]OPTIMIZE                    /OPTIMIZE
/[NO]WARNINGS                    /WARNINGS
```

Prompts

File: file-spec[,...]

Command Parameters

file-spec[,...]

Specifies one or more VAX-11 FORTRAN source programs to be compiled. If you do not specify a file type, the compiler uses the default file type of FOR.

You can specify more than one input file. If you separate the file specifications with commas (,), each file is compiled separately and an object module is produced for each file. If you separate the file specifications with plus signs (+), the files are concatenated and compiled as a single input file, producing one object module and, if /LIST is specified, one listing.

No wild card characters are allowed in the file specifications.

1. Available under separate license.

Command Qualifiers

/CHECK[=(option[,...])]
/NOCHECK

Controls whether the compiler produces extra code to check for program correctness at run time. You can request the following options:

ALL	Provides both BOUNDS and OVERFLOW checks.
NONE	Provides no checking.
[NO]BOUNDS	Controls the production of code to check that all array references are to addresses within the address boundaries specified in the array declaration. The BOUNDS option produces this code.
[NO]OVERFLOW	Enables or disables integer overflow traps. Fixed-point calculations involving BYTE, INTEGER*2, and INTEGER*4 data types are checked for arithmetic overflow when you specify the OVERFLOW option.

By default, if you omit the /CHECK qualifier entirely, the compiler produces code to check only for integer overflow traps (equivalent to /CHECK=NOBOUNDS,OVERFLOW).

However, if you specify /CHECK without options, you obtain both BOUNDS and OVERFLOW checking (equivalent to /CHECK=ALL).

Note that /NOCHECK is equivalent to /CHECK=NONE.

If you specify more than one option, separate them by commas and enclose the list in parentheses.

/CONTINUATIONS=n

Specifies the maximum number of continuation lines to be permitted.

The number of lines, n, is a decimal number from 0 through 99. By default, the compiler accepts a maximum of 19 continuation lines.

/DEBUG[=(option[,...])]
/NODEBUG

Controls whether the compiler makes local symbol table and traceback information available to the debugger and the run time error reporting mechanism.

You can request the following options:

ALL	Provides both local symbol table and traceback information.
NONE	Provide neither local symbol table nor traceback information.

[NO]SYMBOLS	Controls whether the debugger receives local symbol definitions for user-defined variables, arrays, and labels on executable statements. The SYMBOLS option provides this information.
[NO]TRACEBACK	Controls the production of compiler-generated line numbers so that the debugger and the run-time error traceback routine can translate virtual addresses into source program subroutine names and line numbers. The TRACEBACK option produces the line numbers.

By default, if you completely omit the /DEBUG qualifier, the compiler produces only an address correlation table (equivalent to /DEBUG=(NOSYMBOLS,TRACEBACK)). However, if you specify /DEBUG without any options, the default is both SYMBOLS and TRACEBACK (equivalent to /DEBUG=ALL).

Note that /NODEBUG is equivalent to /DEBUG=NONE.

For details on how to debug a VAX-11 FORTRAN program with the VAX-11 Symbolic Debugger, see the VAX-11 FORTRAN User's Guide.

/D_LINES
/NOD_LINES

Indicates whether the compiler reads and compiles lines that have a D in column 1 of the source program. If you specify /D_LINES, lines that have a D in column 1 are compiled.

The default is /NO_DLINES, which means the compiler assumes that lines beginning with a D are comments and does not compile them.

/F77
/NOF77

Controls whether FORTRAN-77 interpretation rules are used for those statements that have a meaning that is incompatible with FORTRAN IV-PLUS.

The default is /F77. If you specify /NOF77, the compiler selects FORTRAN IV-PLUS interpretations in cases of incompatibility.

/G_FLOATING
/NOG_FLOATING

Controls the interpretation of REAL*8, COMPLEX*16, DOUBLE PRECISION and DOUBLE COMPLEX declarations.

The default is /NOG_FLOATING, which causes the compiler to interpret the above declarations as the VAX-11 D_floating data type. If you specify /G_FLOATING, the compiler interprets them as the VAX-11 G_floating data type. See the VAX-11 FORTRAN User's Guide for more details.

/I4
/NOI4

Controls how the compiler interprets INTEGER and LOGICAL declarations that do not specify a length. If you specify /NOI4, the compiler interprets these as INTEGER*2 and LOGICAL*2, respectively.

The default is /I4, which means the compiler interprets these declarations as INTEGER*4 and LOGICAL*4.

/LIST=[file-spec]
/NOLIST

Controls whether a listing file is produced.

If the FORTRAN command is executed from interactive mode, the compiler, by default, does not create a listing file. If the FORTRAN command is executed from batch mode, /LIST is the default; the compiler gives a listing file the same file name as the first input source file and a file type of LIS.

When you specify /LIST, you can control the defaults applied to the output file specification by the placement of the qualifier in the command, as described in Section 5.3.3, "Rules for Entering Output File Qualifiers." The output file type defaults to LIS.

No wild card characters are allowed in the file specification.

/MACHINE_CODE
/NOMACHINE_CODE

Controls whether the listing produced by the compiler includes the machine language code generated by the compiler.

The default is /NOMACHINE_CODE, which omits machine language code in the listing. The /MACHINE_CODE qualifier is ignored if /LIST is not specified, either explicitly or by default.

/OBJECT[=file-spec]
/NOOBJECT

Controls whether the compiler produces an output object module.

By default, the compiler produces an object module that has the same file name as the first input source file and a default file type of OBJ. When you specify /OBJECT, you can control the defaults applied to the output file specification by the placement of the qualifier in the command, as described in Section 5.3.3, "Rules for Entering Output File Qualifiers."

No wild card characters are allowed in the file specification.

/OPTIMIZE
/NOOPTIMIZE

Controls whether the compiler optimizes the compiled program to generate more efficient code.

Use /NOOPTIMIZE in conjunction with the /DEBUG qualifier to link a VAX-11 FORTRAN program with the debugger so that variables always contain their updated values.

/WARNINGS
/NOWARNINGS

Controls whether the compiler produces diagnostic messages for warning conditions.

Use /NOWARNINGS to override the compiler default, which is to issue warning diagnostic messages.

Examples

1.
```
$ FORTRAN SCANLINE
```

   This FORTRAN command compiles the program SCANLINE.FOR and produces an object module SCANLINE.OBJ. If this command is executed in a batch job, the compiler also creates a listing file named SCANLINE.LIS.

2.
```
$ FORTRAN A+B/LIST, C+D/LIST=ALL/OBJECT=ALL
```

   This FORTRAN command requests two separate compilations. For the first compilation, the FORTRAN command concatenates the files A.FOR and B.FOR to produce a single object module named A.OBJ and a listing file named B.LIS that contains the source code from both A.FOR and B.FOR.

   For the second compilation, the FORTRAN command concatenates the files C.FOR and D.FOR and compiles them to produce an object module named ALL.OBJ and a listing named ALL.LIS.

3.
```
$ FORTRAN/NOOPTIMIZE/DEBUG  SCAN/OBJECT=DBGSCAN
$ LINK/DEBUG DBGSCAN
$ RUN DBGSCAN

         VAX-11 DEBUG V2.0

%DEBUG-I-INITIAL, language is FORTRAN, module set to 'SCAN'
DBG>
```

   The FORTRAN command compiles the source program SCAN with the debugger, including both the symbol table information and traceback information; the /OBJECT qualifier requests that the object module be named DBGSCAN. The LINK command links the debugger with the object module and the RUN command executes the image. When the debugger prompts, you can begin entering DEBUG commands.

Transfers control to a labeled statement in a command procedure.

Format

```
GOTO   label

Command Qualifiers                      Defaults

None.                                   None.
```

Prompts

Label: label

Command Parameters

label

Specifies a 1- through 255-alphanumeric character label appearing as the first item on a command line. 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. It must be terminated with a colon (:) and may not contain embedded blanks.

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.

The command interpreter does not check for duplicate labels. The following rules apply:

- 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.

For a description of the GOTO command and examples of its use in command procedures, see the VAX/VMS Guide to Using Command Procedures.

Examples

1.

```
$ IF P1.EQS."HELP" THEN GOTO TELL
$ IF P1.EQS."" THEN GOTO TELL
    .
    .
    .
$ EXIT
$ TELL:
$ TYPE SYS$INPUT
    .
    .
    .
```

The IF command checks the first parameter passed to the command procedure; if this parameter is the string HELP or 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 commmand 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
    .
    .
    .
$ GOTO END
$ CHECK:
    .
    .
    .
$ 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.

Displays on the current default output stream device (SYS$OUTPUT) information available in the system help files or any help library you specify. For more information on creating your own help libraries, see the VAX-11 Utilities Reference Manual.

Format

```
HELP   [keyword ...]

Command Qualifiers                Defaults

/LIBRARY=file-spec                None.
```

Prompts

None.

Command Parameters

keyword ...

Specifies one or more keywords that indicate what information you want. Information is located in a hierarchical manner, depending on the level of information required. The levels are:

1. <NULL> - describes the HELP command, and lists keywords you can specify to obtain information about commands and programs that are documented. Each item in this list is a keyword in the first level of the hierarchy. Most keywords are names of DCL commands, but other information is provided. For example, the keyword SPECIFY is at the first level of a hierarchy of information.

2. Command-name or program-name - describes the command function and format and lists additional information available. This list of information provides keywords for the next level.

3. Command-name or program-name followed by a specific item - provides descriptions of command parameters, or particular keywords or qualifiers.

If you specify an asterisk (*) in place of any keyword, the HELP command displays all information available at that level.

If you specify an ellipsis (...) after any keyword, you obtain everything in the help file at that level.

You may specify percent signs (%) and asterisks (*) in the keywords as wild card characters (see Section 2.1.6).

Command Qualifiers

/LIBRARY=file-spec

Obtains help text from the named library. If specified, this qualifier must precede any optional keywords.

If you omit the device and directory specification, the default is SYS$HELP, the logical name for the location of the system help libraries. The default file type is HLB.

No wild card characters are allowed in the file specification.

Examples

1. **$ HELP ASSIGN**

   The HELP command displays an abstract of the ASSIGN command function, its format, and lists the keyword options you can type to obtain more information.

2. **$ HELP ASSIGN PARAMETERS**

   The HELP command displays a description of the parameters for the ASSIGN command.

3. **$ HELP LEXICAL**

   The HELP command lists the lexical command functions.

4. **$ HELP PRINT/AFTER**

   The HELP command lists the information available about the /AFTER qualifier of the PRINT command.

5. **$ HELP SUBMIT ***

   The HELP command displays information about the SUBMIT command available at the second level, that is, the information that would be displayed if you specified each of the keywords listed when you issued HELP SUBMIT.

6. **$ HELP SET TERMINAL**

   The HELP command lists information about the TERMINAL keyword option of the SET command.

7.
   ```
   $ ASSIGN/USER_MODE  FILE.HLP  SYS$OUTPUT
   $ HELP *...
   ```

   The ASSIGN command defines FILE.HLP as the default output stream for the current process. The HELP command requests that all help text in the default help library SYS$HELP:HELPLIB.HLB be written to FILE.HLP.

8. **$ HELP SPECIFY DELTA_TIME**

   The HELP command informs you how to specify time in the delta time format.

9. **$ HELP ERROR FOR ***

   The HELP command displays the names and descriptions of all VAX-11 FORTRAN run-time error codes.

Tests the value of an expression and executes a command if the test is true. Any arithmetic or logical expression is considered true if the result of the expression is an odd numeric value; an expression is false if the result is an even value.

Format

```
IF expression THEN [$] command
```

Command Qualifiers	Defaults
None.	None.

Prompts

None.

Command Parameters

expression

Defines the test to be performed. The expression can consist of one or more numeric constants, string literals, or symbolic names separated by logical, arithmetic, or string operators.

For a summary of operators and details on the syntax requirements and how to specify expressions, see Section 5.6, "Rules for Forming Expressions."

command

Defines the action to take if the result of the expression is true.

You can specify any valid DCL command following the keyword THEN. Optionally, you can precede the command with a dollar sign.

Description

The IF command provides an effective tool in the development of command procedures. For a description of the IF command and additional examples, see the VAX/VMS Guide to Using Command Procedures.

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 that 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 .NOT.$STATUS THEN EXIT
$ RUN CYGNUS
```

A command procedure uses the SET NOON command to disable error checking by the command procedure. Then, the IF command is used following the execution of a LINK command to test the value of the reserved global symbol $STATUS. If the LINK command returns an error status value, the command procedure exits.

INITIALIZE

Formats and writes a label on a mass storage volume.

Format

```
INITIALIZE   device-name[:]   volume-label


Command Qualifiers[1]                Defaults

/OWNER_UIC=uic                       None.
/PROTECTION=code


Qualifiers for Tapes                 Defaults

/DENSITY=n                           None.
/OVERRIDE=(option[,...])


Qualifiers for Disks                 Defaults

/ACCESSED=n                          /ACCESSED=3
/BADBLOCKS=(list[,...])
/CLUSTER_SIZE=n
/DATA_CHECK[=(option[,...])]         (see text)
/DIRECTORIES=n                       /DIRECTORIES=16
/EXTENSION=n                         /EXTENSION=5
/FILE_PROTECTION=code
/GROUP
/HEADERS=n                           /HEADERS=16
/INDEX=position                      /INDEX=MIDDLE
/MAXIMUM_FILES=n
/[NO]SHARE                           /SHARE
/STRUCTURE=level                     /STRUCTURE=2
/SYSTEM
/USER_NAME=string
/[NO]VERIFIED                        /VERIFIED
/WINDOWS=n                           /WINDOWS=7
```

Prompts

Device: device-name[:]

Label: volume-label

1. For convenience, qualifiers that are applicable only to disks and to tapes are listed separately. All qualifier descriptions appear in alphabetical order, however.

Command 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, this 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 tape volume, you can specify a maximum of 6 alphanumeric characters.

Description

The default format for disk volumes in the VAX/VMS operating system is called the Files-11 Structure Level 2. The default for tape volumes is based on ANSI standard labels, ANSI X3.27-1978, level 3.

The INITIALIZE command can also initialize disk volumes in the Files-11 Structure Level 1 format.

You do not need any special privileges to initialize:

- A blank disk or 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 tape volume that allows write access to your current UIC or that was not protected when it was initialized

In all other cases, you must have the user privilege VOLPRO to initialize a volume.

When the INITIALIZE command initializes a tape volume, it always attempts to read the volume header label. In some cases, a blank tape can cause unrecoverable errors in the command. The symptoms of such an error are:

- The message:

```
%INIT-F-VOLINV, volume is invalid
```

- A runaway tape (this frequently occurs with tapes that have been run through verifying machines). You can only stop a runaway tape by setting the tape drive offline and then putting it back online.

If any such problem occurs, you can successfully initialize a tape by repeating the INITIALIZE command from an account that has the VOLPRO privilege and by specifying the following qualifier in the command:

```
/OVERRIDE=(ACCESSIBILITY,EXPIRATION)
```

This qualifier ensures that the INITIALIZE command will not attempt to verify any labels on the tape.

For examples of initializing and using disks and tapes, see Chapter 3, "Disk and Tape Volumes."

Many of the INITIALIZE command qualifiers allow you to specify parameters that can maximize input/output efficiency. For information on these parameters and a description of the disk structures, see the Introduction to VAX-11 Record Management Services and the VAX-11 Record Management Services Reference Manual.

Command Qualifiers

/ACCESSED=n

Specifies, for disk volumes, the number of directories to be maintained in system space for ready access.

The user privilege OPER is required to use the /ACCESSED qualifier. Legal values for n are 0 through 255. If /ACCESSED is not specified, the INITIALIZE command uses the default value of 3.

/BADBLOCKS=(list[,...])

Specifies, for disk volumes, specific areas on the volume that are faulty. The INITIALIZE command marks the areas as allocated so that no data will be written in them.

You can specify one or more areas, using either or both of the formats shown below. If you specify more than one area, separate the specifications with commas and enclose the list in parentheses.

lbn[:count]	Specify a logical block number on the disk volume, and optionally a count of logical blocks beginning with the logical block specified, to be marked allocated
sector.track.cyl[:count]	Specify a specific sector, track, and cylinder on the disk volume, and optionally a count of blocks, beginning with the first block specified, to be marked allocated

All media supplied by DIGITAL and supported on VAX/VMS, except floppies, TU58 cartridges, and RP04/5/6 disk packs are factory formatted and contain bad block data. The Bad Block Locator (BAD) Utility or the diagnostic formatter ESRAC may be used to refresh the bad block data or construct it for the media exceptions above. The /BADBLOCKS qualifier is only necessary to enter bad blocks that are not indentified in the volume's bad block data.

For information on how to run the BAD Utility, see the VAX-11 Utilities Reference Manual.

/CLUSTER_SIZE=n

Defines, for disk volumes, the minimum allocation unit, in blocks. The maximum size you can specify for a volume is 1/100 the size of the volume; the minimum size you can specify is calculated with the formula:

$$\frac{\text{disk size}}{255*4096}$$

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[,...])]

Defines a default for data check operations following all reads and/or writes to the volume. You can specify either or both of the following options:

READ	Perform checks following all read operations
WRITE	Perform checks following all write operations

If you specify /DATA_CHECK without specifying an option, the system assumes /DATA_CHECK=WRITE. If you do not specify /DATA_CHECK, the system performs no checking as the default. You can override the checking you specify at initialization for disks when you issue a MOUNT command to mount the volume.

If you specify both options, separate them by commas and enclose them in parentheses.

/DENSITY=n

Specifies, for tape volumes, the density in bits per inch (bpi) at which the tape is to be written. You can specify a density of 800, 1600, or 6250, if supported by the tape drive.

If you do not specify a density for a blank tape, the system uses a default density of 1600. If you do not specify a density for a tape that was previously written, the system uses the density at which the tape was last written.

/DIRECTORIES=n

Specifies, for disk volumes, the number of entries to preallocate for user directories.

The legal values are in the range of 16 through 16000; if you do not specify a value, the INITIALIZE command uses the default value of 16.

/EXTENSION=n

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.

You can specify a value in the range of 0 through 65535; if you do not specify a default extension size, the INITIALIZE command uses a value of 5.

/FILE_PROTECTION=code

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 for specifying protection. (These rules are given in Section 5.10.) 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 VAX/VMS system, but is provided to control the process's use of the volume on RSX-11M systems. VAX/VMS always uses the default file protection; the protection can be changed with the SET PROTECTION/DEFAULT command.

/GROUP

Defines a disk volume as a group volume. The owner UIC of the volume defaults to the group number of the user issuing the command and a member number of 0.

If this qualifier is specified in conjunction with the /NOSHARE qualifier, the volume protection is RWED for the system, owner and group. However, the /GROUP qualifier specified alone defines the volume protection as RWED for all user categories.

/HEADERS=n

Specifies, for disk volumes, the number of file headers to be allocated initially for the index file. The minimum value you can specify is 16; the maximum value is the value set with the /MAXIMUM_FILES qualifier.

By default, the INITIALIZE command allocates 16 file headers.

/INDEX=position

Requests, for disk volumes, that the index file for the volume's directory structure be placed in a specific location on the volume.

You can specify one of the following options:

BEGINNING	Place the index file at the beginning of the volume
END	Place the index file at the end of the volume
MIDDLE	Place the index file in the middle of the volume
n	Place the index file at the beginning of the logical block specified by the logical block number n

By default, the INITIALIZE command places the index file in the middle of the volume.

/MAXIMUM_FILES=n

Restricts, for disk volumes, the maximum number of files that the volume can contain, overriding the default value. The default is calculated from the volume size in blocks as follows:

$$\frac{\text{volume size}}{(\text{cluster factor} + 1) * 2}$$

The maximum size you can specify for any volume is:

$$\frac{\text{volume size}}{(\text{cluster factor} + 1)}$$

The minimum value is 0. Note, however, that you should specify a low file maximum only after careful consideration. Once set, the maximum can only be increased by reinitializing the volume.

/OVERRIDE=(option[...])

Requests the INITIALIZE command to ignore data on a tape volume that protects it from being overwritten. You can specify one or both of the following options:

EXPIRATION	Override the expiration date on the volume (the date is indicated by the expiration date of the first file on the volume)
ACCESSIBILITY	Override a nonblank accessibility field in the VOL1 or HDR1 label (this field is never set by VAX/VMS, but may be set by other operating systems)

You must be the owner of the tape volume or have the user privilege to override volume protection (VOLPRO) in order to initialize a tape that has not reached its expiration date or has a nonblank accessibility field.

If you specify more than one option, separate them with commas and enclose the list in parentheses.

/OWNER_UIC=uic

Specifies the user identification code to be assigned ownership of the volume and of system files on the volume. Specify the UIC in the format:

[g,m]

g	is an octal number in range 0 through 377 representing the group number.
m	is an octal number in the range 0 through 377 representing the member number.

The square brackets ([ ]) are required in the UIC specification.

If you do not specify /OWNER_UIC, your current UIC is assigned ownership of the volume.

/PROTECTION=code

Specifies the protection to be applied to the volume. The protection controls who can read, write, create, and delete files on the volume. If you do not specify a protection code, protection defaults to all access to all categories of user. Note that the /GROUP, /SHARE, and /SYSTEM qualifiers can also be used to define protection for disk volumes.

Specify the code according to the standard syntax rules for specifying protection given in Section 5.10. Any attributes not specified default to no access.

When you specify a protection code for an entire disk volume, access type E (execute) indicates create access.

The system only applies read and write access restrictions with respect to tapes; create and delete access are meaningless. Moreover, the system and the owner are always given both read and write access to tapes, regardless of what you specify in a protection code.

/SHARE
/NOSHARE

Controls whether a disk volume is shareable. The protection code for the volume defaults to all types of access for all categories of user. If you specify /NOSHARE, the protection code defaults to no access for group and world.

/STRUCTURE=level

Specifies, for disk volumes, whether the volume should be formatted in Files-11 Structure Level 1 or Structure Level 2. By default, disk volumes are formatted in Files-11 Structure Level 2.

If you specify /STRUCTURE=1, the /CLUSTER_SIZE and /DATA_CHECK qualifiers are not allowed. The default protection for a Structure Level 1 disk is all types of access to system, owner, and group, and read access to all other users.

/SYSTEM

Defines a disk volume as a system volume. The owner UIC of the volume defaults to [1,1] and default protection provides all types of access to the volume to all users.

No user privilege is required to use the /SYSTEM qualifier; however, only users with system UICs can create directories on system volumes.

/USER_NAME=string

Specifies, for disk volumes, a user name of up to 12 characters to be recorded on the volume. If /USER_NAME is not specified, the INITIALIZE command uses the user name under which you logged in.

/VERIFIED
/NOVERIFIED

Indicates, for disk volumes, whether the disk has bad block data on it. The default is /VERIFIED for disks with 4096 blocks or more; the INITIALIZE command assumes that disks contain bad block data and uses the data to mark the bad blocks as allocated. Use /NOVERIFIED to request INITIALIZE to ignore bad block data on the disk. (The default is /NOVERIFIED for disks with less than 4096 blocks.)

/WINDOWS=n

Specifies, for disk volumes, the number of mapping pointers to be allocated for file windows. When a file is opened, the file system uses the mapping pointers to access data in the file. You can specify a value in the range of 7 through 80. The default number of pointers is 7.

Examples

1.
```
$ 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]
$ COPY *.*  TEMP:[ARCHIE]
```

The above sequence of commands shows how to initialize an RK06/RK07 volume for backup. 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 in the CREATE command example. Finally, the COPY command copies the highest existing versions of files on the default disk to the backup disk.

2.
```
$ 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 tape. After allocating a drive, the tape is loaded on the device and the INITIALIZE command writes the label SOURCE on it. Then, the MOUNT command mounts the tape so that files can be written on it.

INQUIRE

Requests interactive assignment of a value for a local or global symbol during the execution of a command procedure.

Format

```
INQUIRE   symbol-name   [prompt-string]
```

Command Qualifiers	Defaults
/GLOBAL	/LOCAL
/LOCAL	/LOCAL
/[NO]PUNCTUATION	/PUNCTUATION

Prompts

None.

Command 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. If the prompt string contains any lowercase characters, multiple blanks or tabs, or an at sign character (@), enclose it in quotation marks (").

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 device SYS$COMMAND. This means that when the INQUIRE command is executed in a command procedure executed interactively, the prompting message is always displayed on the terminal, regardless of the level of nesting of command procedures. When an INQUIRE command is issued in a batch job, the command reads the response from the next line in the command procedure; if procedures are nested, it reads the response from the first level command procedure. If the next line in the batch job command procedure begins with a dollar sign ($), it is another command and the INQUIRE command will not attempt to assign it to the symbol. Rather, the null string is assigned to the symbol and the command procedure execution resumes with the command on the next line following the INQUIRE command.

For more information on how to use commands like this one in command procedures, consult the VAX/VMS Guide to Using Command Procedures.

Command Qualifiers

/GLOBAL

Specifies that the symbol be placed in the global symbol table.

/LOCAL

Specifies that the symbol be placed in the local symbol table for the current command procedure.

This is the default.

/PUNCTUATION
/NOPUNCTUATION

Controls whether or not a colon (:) and a space follow the prompt when it is displayed on the terminal. By default, this punctuation is provided. If you wish 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 IF command tests the value entered. If you enter an odd numeric value or any nonquoted character string that begins with either a T or a Y, the symbol CHECK is considered true and the procedure continues executing. If you enter an even numeric value, any nonquoted character string that begins with an N, an F, or a null string, the symbol is considered false and 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.

Identifies the beginning of a batch job submitted through a card reader.

Format

```
$ JOB   user-name

Command Qualifiers                    Defaults

/AFTER=absolute-time
/CPUTIME=n
/[NO]DELETE                           /DELETE
/NAME=job-name                        /NAME=INPBATCH
/PARAMETERS=(parameter[,...])
/PRIORITY=n
/QUEUE=queue-name[:]                  /QUEUE=SYS$BATCH
/[NO]TRAILING_BLANKS                  /TRAILING_BLANKS
/WSDEFAULT=n
/WSQUOTA=n
```

Prompts

None.

Command Parameters

user-name

Identifies the user name under which the job is to be run. Specify the user name just as you would enter it during the login procedure. All qualifiers you choose to specify must follow the user-name parameter; otherwise the job is not submitted.

Description

All batch jobs submitted to the system through the system card reader must be preceded by a JOB card.

For more information on how to use commands like this one in command procedures, consult the VAX/VMS Guide to Using Command Procedures. The $ is required. The JOB card identifies the user submitting the job, and must be followed by a PASSWORD card giving the password.

The user name and password are validated using 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 account. If a LOGIN.COM file exists, 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.

For more information on how to use commands like this one in command procedures, consult the VAX/VMS Guide to Using Command Procedures.

Command Qualifiers

/AFTER=absolute-time

Requests that the job be held until after a specific time.

Specify the time value according to the rules for entering absolute times (these rules are given in Section 5.8).

If the specified time has already passed, the job is queued for immediate processing.

/CPUTIME=n

Defines a CPU time limit for the batch job. You may' specify a delta time (Section 5.8.2), the value 0, or the words NONE or INFINITE for n.

Use this qualifier to override the base queue value established by the system manager or the value authorized in your user authorization file, when you need less CPU time than authorized. 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 value or the limit specified on the queue. (However, you cannot request more time than permitted by the base limits or your user authorization file.)

/DELETE
/NODELETE

Controls whether the batch input file is saved after the job is processed. By default, the job is deleted after processing. If you specify /NODELETE, the file is saved under the name INPBATCH.COM, by default. If you specify the /NAME qualifier, the file name of the file is the same as the name you specify with /NAME.

/NAME=job-name

Specifies a 1- through 8-alphanumeric character file name string to be used as the job name and as the file name for the batch job log file. By default, the system gives the output log file a file name of INPBATCH.

/PARAMETERS=(parameter[,...])

Specifies from 1 through 8 optional parameters to be passed to the command procedure. The parameters define values to be equated to the symbols named P1, P2, P3, and so on, in the batch job. The symbols are local to the initial input stream.

If you specify more than one parameter, separate them with commas and enclose them in parentheses.

The commas delimit the parameters. To specify a parameter that contains any special characters or delimiters, enclose the parameter in quotation marks (").

The total number of characters enclosed in parentheses to specify the parameters, including the comma (,) and quotation mark (") delimiters, must be less than 95 characters.

/PRIORITY=n

Specifies the priority for the specified job.

The priority, n, must be in the range of 0 through 31, where 0 is the lowest priority and 31 is the highest.

By default, jobs are assigned the same priority as the base priority of your current process; the user privilege OPER is required to set a priority value that is higher than the base priority of your current process.

/QUEUE=queue-name[:]

Specifies the name of a particular batch job queue in which the job is to be entered. If you do not specify /QUEUE, then the job is placed in the default system batch job queue, SYS$BATCH.

/TRAILING_BLANKS
/NOTRAILING_BLANKS

Controls whether input cards in the card deck are read in card image form or if input records are truncated at the last non-blank character. By default, the system does not strip trailing blanks from the 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. You may specify a positive integer in the range 1 through 65535, 0, or the word NONE for n.

Use this qualifier to override the base queue value established by the system manager or the value authorized in your user authorization file, provided you want to impose a lower value. Specify 0 or NONE if you want the working set value defaulted to either your user authorization file or the working set default specified on the queue. However, you may not request a higher value than your default.

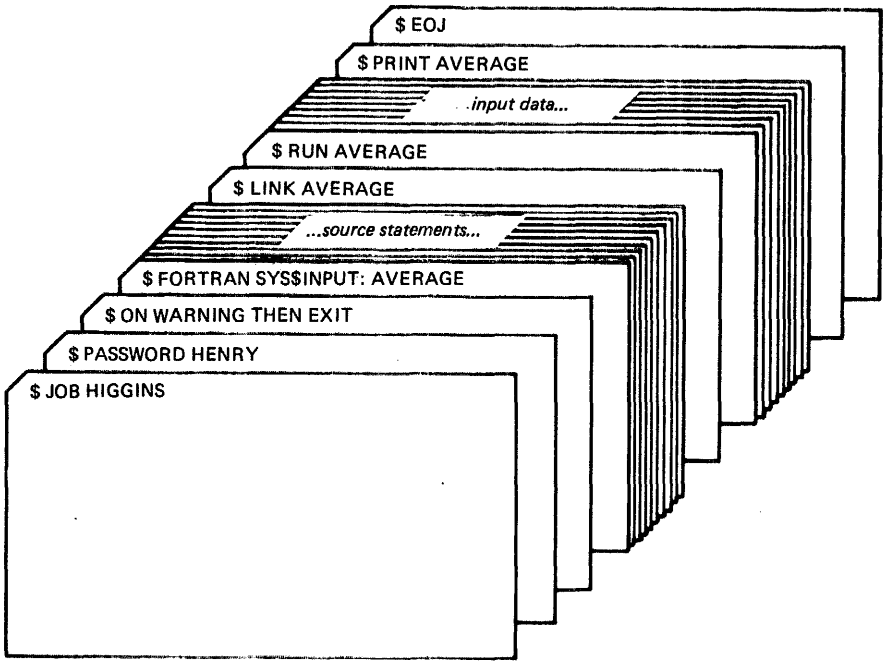
/WSQUOTA=n

Defines the maximum working set size for the batch job. This is the working set quota. You may specify a positive integer in the range 1 through 65535, 0, or the word NONE for n.

Use this qualifier to override the base queue value established by the system manager or the value authorized in your user authorization file, provided you want to impose a lower value. 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. However, you may not request a higher value than your default.

Examples

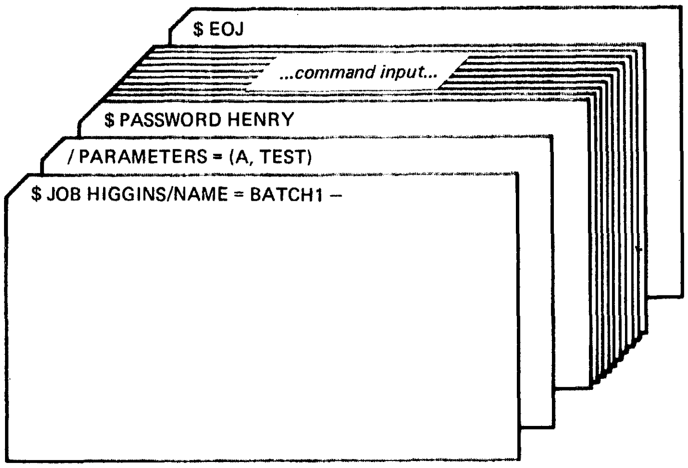
1.



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 the user HENRY'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.

2.



The /NAME qualifier on the JOB card specifies a name for the batch job. When the job completes, the printed log file will be identified as BATCH1.LOG. The JOB card is continued onto a second line with the continuation character (-). The /PARAMETERS qualifier defines P1 as A and P2 as TEST.

Lexical Functions

The command interpreter recognizes a set of functions, called lexical functions, that return information about character strings and attributes of the current process.

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.

Table 1 summarizes the valid functions, their formats, and the information returned by each. Some examples of using lexical functions are given in the VAX/VMS Guide to Using Command Procedures. The syntax requirements for specifying lexical functions are described in detail in Section 5.7, "Rules for Specifying Lexical Functions."

Table 1
Summary of Lexical Functions

Function	Value Returned
F$CVSI(bit-position,count,integer)	Signed value extracted from the specified integer, converted to an ASCII literal
F$CVUI(bit-position,count,integer)	Unsigned value extracted from the specified integer, converted to an ASCII literal
F$DIRECTORY()	Current default directory name string, including brackets
F$EXTRACT(offset,length,string)	Substring beginning at specified offset for length specified of indicated string
F$LENGTH(string)	Length of specified string
F$LOCATE(substring,string)	Relative offset of specified substring within string indicated; or, the length of the string if the substring is not found
F$LOGICAL(logical-name)	Equivalence name of specified logical name (first match found in ordered search of process, group, and system logical name tables); or, a null string if no match is found

(continued on next page)

Table 1 (Cont.)
Summary of Lexical Functions

Function	Value Returned
F$MESSAGE(code)	Message text associated with the specified numeric status code value
F$MODE()	One of the character strings INTERACTIVE or BATCH
F$PROCESS()	Current process name string
F$TIME()	Current date and time of day, in the format dd-mm-yyy hh:mm:ss.cc
F$USER()	Current user identification code (UIC), in the format [g,m]
F$VERIFY(mode)	A numeric value of 1 if verification is set on; a numeric value of 0 if verification is set off The mode parameter specifies the desired verification setting after this function executes; 1 sets verification on while 0 sets verification off

Replaces a module in an object, macro, help, or text library; creates or modifies libraries; inserts, deletes, extracts, or lists the modules or symbols within a library.

For more information on VAX/VMS libraries, see the VAX-11 Utilities Reference Manual.

Format

```
LIBRARY  library-file-spec  [input-file-spec[,...]]
```

Command Qualifiers	Defaults
/COMPRESS[=(option[,...])]	(see text)
/CREATE[=(option[,...])]	(see text)
/CROSS_REFERENCE[=(option[,...])]	(see text)
/DELETE=(module[,...])	
/EXTRACT=(module[,...])	
/FULL	
/[NO]GLOBALS	/GLOBALS
/HELP	/OBJECT
/INSERT	/REPLACE
/[NO]LIST[=file-spec]	/NOLIST
/[NO]LOG	/NOLOG
/MACRO	/OBJECT
/[NO]NAMES	/NONAMES
/OBJECT	/OBJECT
/ONLY=(module[,...]	
/OUTPUT=file-spec	
/REMOVE=(symbol[,...])	
/REPLACE	/REPLACE
/SELECTIVE_SEARCH	
/[NO]SQUEEZE	/SQUEEZE
/TEXT	/OBJECT
/WIDTH=n	

File Qualifiers	Defaults
/MODULE=module-name	None.

Prompts

```
Library:  library-file-spec

File:  input-file-spec[,...]
```

Command Parameters

library-file-spec

Specifies the name of the library you want to create or modify.

Wild card characters are allowed in the library file specification. See Section 2.1.6.

If the file specification does not include a file type, the LIBRARY command assumes a default type of OLB, indicating an object library.

NOTE

Any attempt to modify a library that was created by the VAX/VMS Version 1.0 Librarian, results in an automatic compression into the new format introduced with Version 2.0. The compression occurs prior to the requested modification. (See the /COMPRESS qualifier.)

Furthermore, libraries created prior to Version 2.0 that have not been modified or compressed appear in a different format when listed by the /LIST qualifier.

input-file-spec[,...]

Specifies the names of one or more files that contain modules you want to insert into the specified library.

Whenever you include an input file specification, the LIBRARY command either replaces or inserts the modules contained in the input file(s) into the specified library. The input-file-spec parameter is required when you specify either /REPLACE (the LIBRARY command's default operation) or /INSERT, which is an optional qualifier.

When you use the /CREATE qualifier to create a new library, the input-file-spec parameter is optional. If you include an input file specification with /CREATE, the LIBRARY command first creates a new library, and then inserts the contents of the input file(s) into the library.

Note that the /EXTRACT qualifier does not accept an input file specification.

If you specify more than one input file, separate the file specifications with a comma (,). The LIBRARY command will then insert the contents of each file into the specified library.

If any file specification does not include a file type, the LIBRARY command assumes a default file type of OBJ, designating an object library. You can control the default file type by specifying the appropriate qualifier as indicated below:

Qualifier	**Default File Type**
/HELP	HLP
/MACRO	MAR
/OBJECT	OBJ
/TEXT	TXT

Note also that the file type you specify on the library-file-spec parameter can affect the default file type of the input file specification, provided the /CREATE qualifier is not being issued. For example, if the library file type is HLB, MLB, OLB, or TLB, the input file type default is HLP, MAR, OBJ, or TXT, respectively.

Wild card characters are allowed in the input file specification(s). See Section 2.1.6.

Description

Libraries are files that contain one or more directories pointing to the locations of individual modules. The LIBRARY command creates libraries and modifies their contents. You use DCL commands to manipulate a library in its entirety; for example, the DELETE, COPY, and RENAME commands delete, make copies of or rename libraries, respectively.

The LIBRARY command distinguishes four types of libraries:

- Object module libraries contain frequently called routines. You can use object module libraries as input to the linker. The linker searches the object module library whenever it encounters a reference it cannot resolve from the specified input files.

- Macro libraries contain macro definitions. You can use macro libraries as input to the assembler. The assembler searches the macro library whenever it encounters a macro that is not defined in the input file.

- Help libraries contain help text. You can retrieve help messages by calling the appropriate library procedures from your program. See the VAX-11 Utilities Reference Manual for information about calling library procedures.

- Text libraries contain any sequential record file that you want to retrieve as data for your program. You can retrieve text from text libraries by calling the appropriate library procedures from your program. See the VAX-11 Utilities Reference Manual for information about calling library procedures.

All libraries contain a directory called a module name table (MNT) that names the modules in the library. Object module libraries also contain a global symbol table (GST) that is a list of the global symbols defined in each of the modules in the library. When the LIBRARY command adds a module to a library, it catalogs the module by its module name, rather than the input file specification. The only exception to this procedure occurs with text libraries, where the name of the input file containing the text automatically becomes the module name.

When using the LIBRARY command, you can specify qualifiers that request more than one function in a single command, with some restrictions. Generally, you cannot specify multiple qualifiers that request incompatible functions. The qualifiers that perform library functions, related qualifiers, and qualifier incompatibilities are summarized in Table 2.

Table 2
LIBRARY Command Qualifiers

Qualifier	Related Qualifiers	Incompatible Qualifiers
/COMPRESS	/OUTPUT	/CREATE, /EXTRACT
/CREATE [1]	/SQUEEZE [2], /GLOBALS [3], /SELECTIVE_SEARCH [3]	/COMPRESS, /EXTRACT
/CROSS_REFERENCE	/ONLY	/EXTRACT
/DELETE	—	/EXTRACT
/EXTRACT	/OUTPUT	/COMPRESS, /CREATE, /DELETE, /INSERT, /LIST, /REMOVE, /REPLACE
/INSERT	/SQUEEZE [2], /GLOBALS [3], /SELECTIVE_SEARCH [3]	/EXTRACT
/LIST	/FULL, /NAMES [3], /ONLY	/EXTRACT
/MODULE [4]	/TEXT	/EXTRACT, /DELETE, /REMOVE
/REMOVE [3]	—	/EXTRACT
/REPLACE	/SQUEEZE [2], /GLOBALS [3], /SELECTIVE_SEARCH [3]	/EXTRACT

1. The /CREATE, /INSERT, and /REPLACE qualifiers are not incompatible; however, if you specify more than one, then /CREATE takes precedence over /INSERT, and /INSERT takes precedence over /REPLACE. The related qualifiers for /CREATE are applicable only if you enter one or more input files.

2. Indicates a qualifier that applies only to macro libraries

3. Indicates a qualifier that applies only to object libraries

4. Indicates a qualifier that applies only to text libraries

Command Qualifiers

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

Requests the LIBRARY command to perform either of the following functions:

- Recover unused space in the library resulting from module deletion, or:
- Reformat a library created by the VAX/VMS Version 1.0 Librarian into a Version 2.0 format.

When you specify /COMPRESS, the LIBRARY command by default creates a new library with a version number one higher than the existing library. Use the /OUTPUT qualifier to specify an alternate name for the compressed library.

Specify one or more of the following options to increase or decrease the size of the library, overriding the values specified when the library was created:

BLOCKS:n	Specify the number of 512-byte blocks to be allocated for the library
GLOBALS:n	Specify the maximum number of global symbols the library can contain (for object module libraries only)
KEYSIZE:n	Change the maximum length of a module name or global symbol
MODULES:n	Specify the maximum number of modules or macros the library can contain

If you specify more than one option, separate them with commas and enclose the list in parentheses.

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

Requests the LIBRARY command to create a new library. When you specify /CREATE, you can optionally specify a file or a list of files that contains modules to be placed in the library.

By default, the LIBRARY command creates an object module library; specify /MACRO, /HELP, or /TEXT to change the default library type.

Specify one or more of the following options to control the size of the library, overriding the system defaults:

BLOCKS:n	Specify the number of 512-byte blocks to be allocated for the library. By default, the LIBRARY command allocates 100 blocks for a new library.
GLOBALS:n	Specify the maximum number of global symbols the library can contain initially. By default, the LIBRARY command sets a maximum of 128 global symbols for an object module library. (Macro, help, and text libraries do not have a global symbol directory; therefore, the maximum for these libraries defaults to 0.)

KEYSIZE:n	Define the maximum name length of modules and global symbols. By default the LIBRARY command limits the names of object, macro, and text modules and global symbols to 31 characters. The limit for help modules is 15 characters. When you specify a keysize value, remember that VAX-11 MACRO and the linker will not accept module names or global symbol names in excess of 31 characters.
MODULES:n	Specify the maximum number of modules the library can contain. By default, the LIBRARY command sets an initial maximum of 512 modules for an object module library and 256 modules for all other libraries. An index in a library can grow past its initial allocation. However, for optimum performance, it is best to allocate the maximum number of modules you expect to use.

If you specify more than one option, separate them with commas and enclose the list in parentheses.

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

Requests a cross reference listing of an object library.

If you omit this qualifier, cross reference listings are not provided. However, if you specify /CROSS_REFERENCE without specifying an option, you will obtain cross reference listings by default that contain only symbols by name and symbols by value.

You may specify one or more of the following options:

ALL	Specifies that all types of cross references are desired
MODULE	Specifies a cross reference of both the global symbol references in the module and the global symbol definitions
NONE	Specifies that no cross reference listing is desired
SYMBOL	Provides a cross reference by symbol name
VALUE	Provides a cross reference of symbols by value

If you specify more than one option, separate them with commas and enclose the list in parentheses.

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

Requests the LIBRARY command to delete one or more modules from a library. You must specify the names of one or more modules to be deleted from the library. If you specify more than one module, separate each with commas and enclose the list in parentheses.

Wild card characters are allowed in the module specification. See Section 2.1.6.

If you specify the /LOG qualifier in conjunction with /DELETE, the LIBRARY command issues the message:

```
%LIBRAR-S-DELETED, MODULE module-name DELETED FROM library-name
```

The LIBRARY command physically removes modules from a library.

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

Copies one or more modules from an existing library into a new file. If you specify more than one module, separate the module names with commas and enclose the list in parentheses.

Wild card characters are allowed in the module specification. See Section 2.1.6.

If you specify the /OUTPUT qualifier in conjunction with /EXTRACT, the LIBRARY command writes the output into the specified output file. If you specify /EXTRACT and do not specify /OUTPUT, the LIBRARY command writes the file into a file that has the same file name as the library and a file type of OBJ, MAR, HLP, or TXT depending on the type of library.

/FULL

Requests a full description of each module in the module name table. Use this qualifier in conjunction with the /LIST qualifier to request a list of each library module in the format:

```
module-name Ident nn Inserted dd-mmm-yyyy hh:mm:ss n symbols
```

/GLOBALS
/NOGLOBALS

Controls, for object module libraries, whether the names of global symbols in modules being inserted in the library are included in the global symbol table.

By default, the LIBRARY command places all global symbol names in the global symbol table. Use /NOGLOBALS when you do not want global symbol names in the global symbol table.

/HELP

Indicates that the library is a help library. When you specify the /HELP qualifier, the library file type defaults to HLB and the input file type defaults to HLP.

For information on how to create help files, see the VAX-11 Utilities Reference Manual.

/INSERT

Requests the LIBRARY command to add the contents of one or more files to an existing library. If an object module file specified as input consists of concatenated object modules, the LIBRARY command creates a separate entry for each object module in the file; each module name table entry reflects an individual module name. If a macro or help file specified as input contains more than one definition, the LIBRARY command creates a separate entry for each one, naming the module name table entries according to the names specified on the .MACRO directives or in the HELP format.

When the LIBRARY command inserts modules into an existing library, it checks the module name table before inserting each module. If a module name or global symbol name already exists in the library, the command issues an error message and does not add the module to the library.

To insert or replace a module in a library regardless of whether there is a current entry with the same name, use the /REPLACE qualifier.

/LIST[=file-spec]
/NOLIST

Controls whether or not the LIBRARY command creates a listing of the contents of the library.

By default, no listing is produced. If you specify /LIST without a file specification, the LIBRARY command writes the output file to the current SYS$OUTPUT device. If you include a file specification that does not have a file type, the LIBRARY command uses the default file type of LIS.

No wild card characters are allowed in the file specification.

If you specify /LIST in conjunction with qualifiers that perform additional operations on the library, the LIBRARY command creates the listing after completing all other requests; thus, the listing reflects the status of the library after all changes have been made.

When you specify /LIST, the LIBRARY command provides, by default, the following information about the library:

```
Directory of OBJECT library  DBB0:[LIBRAR]LIBRAR.OLB;1 on 14-NOV-1979 10:08:28
Creation date:  12-NOV-1979 19:40:36      Creator:  VAX-11 Librarian V02.00
Revision date:  14-NOV-1979 16:04:58      Library format:  1.1
Number of modules:      15                Max. key length:  31
Other entries:          73                Preallocated index blocks:     35
Recoverable deleted blocks:     15        Total index blocks used:       12
```

/LOG
/NOLOG

Controls whether the LIBRARY command verifies each library operation. If you specify /LOG, the LIBRARY command displays the module name, followed by the library operation performed, followed by the library file specification.

/MACRO

Indicates that the library is a macro library. When you specify /MACRO, the library file type defaults to MLB and the input file type defaults to MAR.

/NAMES
/NONAMES

Controls, when /LIST is specified for an object module library, whether the LIBRARY command lists the names of all global symbols in the global symbol table as well as the module names in the module name table.

The default is /NONAMES, which does not list the global symbol names. If you specify /NAMES, each module entry name is displayed in the format:

```
Module module-name
global-symbol      global-symbol      global-symbol      global-symbol
      .                  .                  .                  .
      .                  .                  .                  .
      .                  .                  .                  .
```

If the library is a macro, help, or text library and you specify /NAMES, no symbol names are displayed.

/OBJECT

Indicates that the library is an object module library. This is the default condition. The LIBRARY command assumes a library file type of OLB and an input file type of OBJ.

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

Specifies the individual modules on which the LIBRARY command may operate. When you use the /ONLY qualifier, the LIBRARY command lists or cross references only those modules specified.

If you specify more than one module, separate the module names with commas and enclose the list in parentheses.

Wild card characters are allowed in the module name specification(s). See Section 2.1.6.

/OUTPUT=file-spec

Specifies, when used with the /EXTRACT, /COMPRESS, or /CROSS_REFERENCE qualifiers, the file specification of the output file.

For /EXTRACT, the output file contains the modules extracted from a library; for /COMPRESS, the output file contains the compressed library; for /CROSS_REFERENCE the output file contains the cross reference listing.

No wild card characters are allowed in the file specification.

If you omit the file type in the file specification, a default is used depending on the library function qualifier and, in some cases, the library type qualifier as shown below:

Qualifier	**Library Type Qualifier**	**Default File Type**
/COMPRESS	/HELP	HLB
	/MACRO	MLB
	/OBJECT	OLB
	/TEXT	TLB
/CROSS_REFERENCE	---	LIS
/EXTRACT	/HELP	HLP
	/MACRO	MAR
	/OBJECT	OBJ
	/TEXT	TXT

/REMOVE=(symbol[....])

Requests the LIBRARY command to delete global symbol entries from the global symbol table in an object library. If you specify more than one symbol, separate them with commas and enclose the list in parentheses.

If you want to verify the names of the deleted global symbols, you must also specify the /LOG qualifier.

Wild card characters are allowed in the symbol specifications. See Section 2.1.6.

/REPLACE

Requests the LIBRARY command to replace one or more existing library modules with the modules specified in the input file. The LIBRARY command first deletes any existing library modules with the same name as the modules in the input file. Then, the new version of the module is inserted in the library. If any modules contained in the input file do not have a corresponding module in the library, the LIBRARY command inserts the new modules in the library.

This is the LIBRARY command's default operation. If you specify an input file parameter, the library command either replaces or inserts the contents of the input file into the library. If you use the /LOG qualifier with the /REPLACE qualifier, the LIBRARY command displays, in the following form, the names of each module that it replaces or inserts.

```
%LIBRAR-S-REPLACED, MODULE module-name REPLACED IN library-file-spec

%LIBRAR-S-INSERTED, MODULE module-name INSERTED IN library-file-spec
```

/SELECTIVE_SEARCH

Defines the input files being inserted into a library as candidates for selective searches by the linker. If you specify /SELECTIVE_SEARCH, the linker selectively searches the modules when the library is specified as a linker input file; the linker only includes the global symbol(s) in the module(s) referenced by other modules in the symbol table of the output image file.

/SQUEEZE
/NOSQUEEZE

Controls whether the LIBRARY command compresses individual macros before adding them to a macro library. When you specify /SQUEEZE, which is the default, trailing blanks, trailing tabs, and comments are deleted from each macro before insertion in the library.

Use /SQUEEZE in conjunction with the /CREATE, /INSERT, and /REPLACE qualifiers to conserve space in a macro library. If you want to retain the full macro, specify /NOSQUEEZE.

/TEXT

Indicates that the library is a text library. When you use the /TEXT qualifier, the library file type defaults to TLB and the input file type defaults to TXT. For more information on text libraries, see the VAX-11 Utilities Reference Manual.

/WIDTH=n

Controls the screen display width (in characters) when listing global symbol names. Specify the /WIDTH qualifier with the /NAMES qualifier to limit the line length of the /NAMES display.

The default display width is the width of the listing device. The maximum width is 132.

File Qualifiers

/MODULE=module-name

Specifies the module name of a text module. Unlike help, object, and macro libraries, text libraries use the file name from the input-file-spec parameter as the module name. If you want the module to have a different name from the input file name, use the /MODULE qualifier to identify the added module.

No wild card characters are allowed in the module name.

You can also use the /MODULE qualifier to enter a text module interactively.

If you specify SYS$INPUT as the input file specification and also issue the /MODULE qualifier, the LIBRARY command includes the text you enter from the console in the specified library module. (To terminate the console input, enter a CTRL/Z.)

Examples

1.
```
$ LIBRARY/CREATE TESTLIB ERRMSG,STARTUP
```

The LIBRARY command creates an object module library named TESTLIB.OLB and places the modules ERRMSG.OBJ and STARTUP.OBJ in the library.

2.
```
$ LIBRARY/INSERT TESTLIB SCANLINE
$ LINK TERMTEST TESTLIB/LIBRARY
```

The LIBRARY command adds the module SCANLINE.OBJ to the library TESTLIB.OLB. The library is specified as input to the linker by using the /LIBRARY qualifier on the LINK command. If the module TERMTEST.OBJ refers to any routines or global symbols not defined in TERMTEST, the linker will search the global symbol table of library TESTLIB.OLB to resolve the symbols.

3.
```
$ LIBRARY/EXTRACT=(ALLOCATE,APPEND)/OUTPUT=MYHELP -
$_SYS$HELP:HELPLIB.HLB
```

The LIBRARY command specifies that the modules ALLOCATE and APPEND be extracted from the help library HELPLIB.HLB and output to the file MYHELP.HLP.

4.
```
$ LIBRARY/CROSS_REFERENCE=ALL/OUTPUT=SYS$OUTPUT LIBRAR
```

The LIBRARY command requests a cross reference listing of the object library LIBRAR.OLB. The cross reference listing is output on the terminal. The listing includes cross references by symbol, by value, and by module.

5. `$ LIBRARY/REMOVE=(LIB_EXTRCT_MODS,LIB_INPUT_MAC)/LOG LIBRAR`

   The LIBRARY command requests the removal of the global symbols LIB_EXTRCT_MODS and LIB_INPUT_MAC from the object library LIBRAR.OLB. The /LOG qualifier requests that the removal of the symbols be confirmed by messages.

6.
   ```
   $ LIBRARY/MACRO/CREATE=(BLOCKS:40,MODULES:100) MYMAC TEMP
   $ MACRO MYMAC/LIBRARY,CYGNUS/OBJECT
   ```

   The LIBRARY command creates a macro library named MYMAC.MLB from the macros in the file TEMP.MAR. The new library has room for 100 modules in a 40-block file. If the input file contains multiple macros, each macro is entered in the new library.

   The MACRO command assembles the source file CYGNUS.MAR; the /LIBRARY qualifier specifies the library MYMAC.MLB as an input file. If the source file CYGNUS contains any macro calls not defined within the file, the assembler searches the library.

7. `$ LIBRARY/LIST=MYMAC.LIS/FULL MYMAC.MLB`

   The LIBRARY command requests a full listing of the macro library MYMAC; the output is written to a file named MYMAC.LIS.

8. `$ LIBRARY/INSERT/TEXT TSTRING SYS$INPUT/MODULE=TEXT1`

   The LIBRARY command inserts a module named TEXT1 into the text library TSTRING.TLB. The input is taken from SYS$INPUT.

9. `$ LIBRARY/LIST/NAMES/ONLY=$ONE/WIDTH=80 SYMBOLIB`

   The LIBRARY command requests a full listing of the module $ONE, contained in the object library SYMBOLIB.OLB. The /WIDTH qualifier requests that the display be limited to 80 characters per line, so that the full listing will not be truncated on the video terminal.

Creates or modifies an RSX-11M object module library or an RSX-11M MACRO library; or inserts, deletes, replaces, or lists modules, macros, or global symbol names in a library. The /RSX11 qualifier is required.

For more information on RSX-11M libraries, see the RSX-11 Utilities Manual.

Format

```
LIBRARY  library-file-spec  [input-file-spec[,...]]


    Additional
Command Qualifiers                Defaults

/COMPRESS[=(option[,...])]        /REPLACE
/CREATE[=(option[,...])]          /REPLACE
/DELETE=(module[,...])            /REPLACE
/EXTRACT[=(module[,...])]
/FULL
/[NO]GLOBALS                      /GLOBALS
/INSERT                           /REPLACE
/[NO]LIST[=file-spec]             /NOLIST
/MACRO                            /OBJECT
/[NO]NAMES                        /NONAMES
/OBJECT                           /OBJECT
/OUTPUT=file-spec
/REMOVE=(symbol[,...])
/REPLACE                          /REPLACE
/SELECTIVE_SEARCH
/SQUEEZE
```

Prompts

Library: library-file-spec

File: input-file-spec[,...]

Command Parameters

library-file-spec

Specifies the name of the library to be created or modified.

No wild card characters are allowed in the library file specification. If the file specification does not include a file type, the LIBRARY/RSX11 command assumes a default file type of OLB if /OBJECT is specified either explicitly or by default; or a default file type of MLB if /MACRO is specified.

input-file-spec[,...]

Specifies the names of one or more files that contain modules to be inserted in the specified library. This parameter is required when you specify /INSERT or /REPLACE; it is optional when you specify /CREATE. Note that the default operation for the LIBRARY/RSX11 command is /REPLACE; if you do not specify a qualifier that requests a specific operation, you must enter the input-file-spec parameter.

If you do not specify any of the qualifiers /REPLACE, /INSERT, or /CREATE, the input-file-spec parameter is invalid.

If you specify more than one file, you can separate the file specifications with either plus signs (+) or commas (,). In either case, the contents of each file are inserted in the specified library.

No wild card characters are allowed in the input file specifications.

If any file specification does not include a file type, the LIBRARY/RSX11 command assumes a default file type of OBJ when /OBJECT is specified either implicitly or by default, and a file type of MAC when /MACRO is specified.

Description

Libraries are files that contain one or more entries, or modules, of a similar type; and directories, or tables, that indicate the locations of individual modules within the library. There are two types of libraries:

- Object module libraries that catalog frequently called routines; you can use object module libraries as input to the RSX-11M Task Builder. The RSX-11M Task Builder searches the object module library when it encounters a reference it cannot resolve from the input files specified.

- Macro libraries that catalog macro definitions; you can use macro libraries as input to the assembler. The assembler searches the macro library when it encounters a .MCALL assembler directive.

Both object module libraries and macro libraries have a directory called a module name table (MNT) that lists the modules (or, in the case of macro libraries, the macros) in the library. An object module library also contains a global symbol table (GST) that lists the global symbols defined in each of the modules in the library.

When the LIBRARY/RSX11 command adds a file to an object module library, it catalogs the module according to its module name, and not by the file name of the file containing the module.

The LIBRARY/RSX11 command creates libraries and modifies their contents. You can use DCL commands to manipulate a library in its entirety, for example, the DELETE, COPY, and RENAME commands can delete, make copies of, or rename libraries, respectively.

When you use the LIBRARY/RSX11 command, you can specify qualifiers that request more than one function in a single command, with some restrictions. The qualifiers that perform LIBRARY functions, related qualifiers, and qualifier incompatibilities are summarized in Table 3.

Table 3
LIBRARY/RSX11 Command Qualifiers

Qualifier	Related Qualifiers	Incompatible Functions
/COMPRESS	/OUTPUT	/CREATE, /EXTRACT
/CREATE[1]	/SQUEEZE[2], /GLOBALS[3], /SELECTIVE_SEARCH[3]	/COMPRESS, /EXTRACT
/DELETE	---	/EXTRACT
/EXTRACT	/OUTPUT	/COMPRESS, /CREATE, /DELETE, /INSERT, /LIST, /REMOVE, /REPLACE
/INSERT	/SQUEEZE[2], /GLOBALS[3], /SELECTIVE_SEARCH[3]	/EXTRACT
/LIST	/FULL, /NAMES[3]	/EXTRACT
/REMOVE[3]	---	/EXTRACT
/REPLACE	/SQUEEZE[2], /GLOBALS[3], /SELECTIVE_SEARCH[3]	/EXTRACT

1. The /CREATE, /INSERT, and /REPLACE qualifiers are not incompatible; however, if more than one is specified, /CREATE takes precedence over /INSERT and /INSERT takes precedence over /REPLACE. The related qualifiers for /CREATE are applicable only if you enter one or more input files.

2. Indicates a qualifier that applies only to macro libraries

3. Indicates a qualifier that applies only to object module libraries

Additional Command Qualifiers

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

Requests the LIBRARY/RSX11 command to recover unused space in the library resulting from module deletion. When you specify /COMPRESS, the LIBRARY/RSX11 command by default creates a new library with a version number one higher than the existing library. Use the /OUTPUT qualifier to specify an alternate name for the compressed library.

If you omit this qualifier, the default is /REPLACE.

You can optionally specify one or more of the following options to increase or decrease the size of the library, overriding the values specified when the library was created:

BLOCKS:n	Specify the number of 512-byte blocks to be allocated for the library
GLOBALS:n	Specify the maximum number of global symbols the library can contain (for object module libraries only)
MODULES:n	Specify the maximum number of modules or macros the library can contain

If you specify more than one option, separate them with commas or plus signs and enclose the list in parentheses.

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

Requests the LIBRARY/RSX11 command to create a new library. When you specify /CREATE, you can optionally specify a file or a list of files that contain modules to be placed in the library.

If you omit this qualifier, the default is /REPLACE.

By default, the LIBRARY/RSX11 command creates an object module library; specify /MACRO to indicate that the library is a macro library.

Specify one or more of the following options to control the size of the library, overriding the system defaults:

BLOCKS:n	Specify the number of 512-byte blocks to be allocated for the library. By default, the LIBRARY/RSX11 command allocates 100 blocks for a new library.
GLOBALS:n	Specify the maximum number of global symbols the library can contain. By default, the LIBRARY/RSX11 command sets a maximum of 128 global symbols for an object module library. (A macro library does not have a global symbol directory; therefore the maximum for macro libraries defaults to 0.)
MODULES:n	Specify the maximum number of modules the library can contain. By default, the LIBRARY/RSX11 command sets a maximum of 512 modules for an object module library and 256 modules for a macro library. If you specify the MODULES option, the maximum value allowed for n is 4096.

If you specify more than one option, separate them with commas and enclose the list in parentheses.

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

Requests the LIBRARY/RSX11 command to delete one or more modules from a library. You must specify the names of one or more modules to be deleted from the library. If you specify more than one module, separate them with commas and enclose the list in parentheses. No wild card characters are allowed in the module specification(s).

When the LIBRARY/RSX11 command deletes modules from a library, it issues the message:

```
MODULES DELETED:
```

Then, it lists the names of modules it has successfully deleted.

The LIBRARY/RSX11 command does not physically remove a module from a library, but rather deletes its entry in the module name table. Use the /COMPRESS qualifier to compress a library from which modules have been deleted.

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

Copies one or more modules from an existing library into a new file. If you specify more than one module, separate the module names with commas and enclose the list in parentheses. No wild card characters are allowed in the module specification(s).

If you specify the /OUTPUT qualifier in conjunction with /EXTRACT, the LIBRARY/RSX11 command writes the output into the file specified by the /OUTPUT qualifier. If you specify /EXTRACT and do not specify /OUTPUT, the LIBRARY/RSX11 command writes the file into a file that has the same file name as the library and a file type of OBJ, or MAC, depending on the /OBJECT and /MACRO qualifiers.

If you specify /EXTRACT, the module name on the /EXTRACT qualifier is optional: all modules in the specified library are concatenated into a single file with a file type of OBJ or MAC, depending on the /OBJECT and /MACRO qualifiers.

/FULL

Requests a full description of each module in the module name table. Use this qualifier in conjunction with the /LIST qualifier to request each module in the library be listed in the format:

```
entry SIZE:nnnnn INSERTED:dd-mmm-yyyy IDENT:nn
```

/GLOBALS
/NOGLOBALS

Controls, for object module libraries, whether the names of global symbols in modules being inserted in the library are included in the global symbol table.

By default, the LIBRARY/RSX11 command places all global symbol names in the global symbol table. Use /NOGLOBALS when you do not want the global symbol names in the global symbol table.

/INSERT

Requests the LIBRARY/RSX11 command to add the contents of one or more files to an existing library. If an object module file specified as input consists of concatenated object modules, the LIBRARY/RSX11 command creates a separate entry for each object module in the file; each module name table entry reflects an individual module name. If a macro file specified as input contains more than one macro definition, the LIBRARY command creates a separate entry for each macro, naming the module name table entries according to the names specified on the .MACRO directives.

When the LIBRARY/RSX11 command inserts modules into an existing library, it checks the module name table before inserting each module. If a module name, macro name, or global symbol name already exists in the library, the command issues an error message and does not add the module to the library. One or more modules may be successfully inserted before the error occurs.

To insert or replace a module in a library regardless of whether there is a current entry with the same name, use the /REPLACE qualifier. The default is /REPLACE.

/LIST[=file-spec]
/NOLIST

Controls whether or not the LIBRARY/RSX11 command creates a listing of the contents of the library.

The default is /NOLIST. If you specify /LIST without including a file specification, the LIBRARY/RSX11 command writes the output file to the current SYS$OUTPUT device. If you include a file specification that does not have a file type, the LIBRARY/RSX11 command uses the default file type of LST.

If you specify /LIST in conjunction with qualifiers that perform additional operations on the library, the LIBRARY/RSX11 command creates the listing after completing all the other requests; thus the listing reflects the status of the library after all changes have been made.

No wild card characters are allowed in the file specification.

When you specify /LIST, the LIBRARY/RSX11 command provides, by default, the following information about the library:

DIRECTORY OF FILE file-spec

File name, file type, and version number of the library being listed.

library-type LIBRARY CREATED BY: LBR vvvvvv

The type of library (OBJECT or MACRO) and the version number of the librarian that created the library.

LAST INSERT OCCURRED dd-mmm-yyyy AT hh:mm:ss

The date and time at which the last insertion was made.

MNT ENTRIES ALLOCATED: nn; AVAILABLE: mm

The current status of the module name table: the maximum number of entries that can be entered in the table (nn) and the number of entries that are unused (mm).

EPT ENTRIES ALLOCATED: nn; AVAILABLE: mm

The current status of the entry point table: the maximum number of entries that can be entered in the table (nn), and the number of entries that are unused (mm). For a macro library, both values are always 0.

FILE SPACE AVAILABLE: nnnnn WORDS

The amount of space available in the library for new files.

RECOVERABLE DELETED SPACE: nnnnn WORDS

The amount of space occupied by modules whose entries have been deleted from the module name table. To recover this space, use the /COMPRESS qualifier.

```
module
module
  .
  .
  .
```

The names of all the entries in the module name table.

If you specify /LIST, you can also specify /FULL and /NAMES to request additional information in the listing.

/MACRO

Indicates that the library is a macro library containing Macros for the MACRO-11 assembler.

The input file type defaults to MAC.

/NAMES
/NONAMES

Controls, when /LIST is specified for an object module library, whether the LIBRARY/RSX11 command lists the names of all global symbols in the global symbol table as well as the module names in the module name table.

The default is /NONAMES; if you specify /NAMES, each module entry name is displayed in the format:

```
** MODULE:entry-name

  symbol   symbol   symbol   symbol   symbol   symbol
    .        .        .        .        .        .
    .        .        .        .        .        .
    .        .        .        .        .        .
```

If the library is a macro library and you specify /NAMES, no symbol names are displayed.

/OBJECT

Indicates that the library is an object module library in RSX-11M format. This is the default.

/OUTPUT=file-spec

Specifies, when the /EXTRACT or /COMPRESS qualifiers are specified, the file specification of the output file. No wild card characters are allowed in the file specification.

For /EXTRACT, the output file contains the modules extracted from a library; for /COMPRESS, the output file contains the compressed library.

/REMOVE=(symbol[,...])

Requests the LIBRARY/RSX11 command to delete entries for one or more global symbols from the global symbol table.

When you specify /REMOVE, the LIBRARY/RSX11 command displays the message:

```
GLOBAL SYMBOLS DELETED:
```

Then, it displays the names of global symbols it successfully deleted.

/REPLACE

Requests the LIBRARY/RSX11 command to replace one or more existing modules in a library with the modules in the input file or files specified. The LIBRARY/RSX11 command first deletes an existing entry, if any, for each module or macro in the input file(s) and the corresponding global symbols, then inserts the new module or macro in the library.

This is the default operation; if you specify an input file parameter and do not specify /CREATE, /INSERT, or /REPLACE, the LIBRARY/RSX11 command replaces an existing module(s) in the file with the modules in the files specified.

When you use the /REPLACE function, the LIBRARY/RSX11 command displays the names of each module replaced, in the format:

```
MODULE "module-name" REPLACED
```

/SELECTIVE_SEARCH

Defines the input files being inserted into a library as candidates for selective searches by the linker. If you specify /SELECTIVE_SEARCH, the modules are selectively searched by the linker when the library is specified as a linker input file: only the global symbols in the module(s) that are referenced by other modules are included in the symbol table of the output image file.

/SQUEEZE

Requests that the LIBRARY/RSX11 command compress individual macros before adding them to a macro library. When you specify /SQUEEZE, trailing blanks, trailing tabs, and comments are deleted from each macro before insertion in the library.

Use /SQUEEZE in conjunction with the /CREATE, /INSERT, and /REPLACE qualifiers to conserve space in a macro library. By default, macros are compressed.

Examples

1. `$ LIBRARY/RSX11/CREATE TESTLIB ERRMSG,STARTUP`

   The LIBRARY/RSX11 command creates an object module library named TESTLIB.OLB and places the modules ERRMSG.OBJ and STARTUP.OBJ in the library.

2. **$ LIBRARY/RSX11/INSERT TESTLIB SCANLINE**
   $ LINK/RSX11 TERMTEST,TESTLIB/LIBRARY

   The LIBRARY/RSX11 command adds the module SCANLINE.OBJ to the library TESTLIB.OLB. The library is specified as input to the Task Builder by using the /LIBRARY qualifier on the LINK/RSX11 command. If the module TERMTEST.OBJ refers to any routines or global symbols not defined in TERMTEST, the RSX-11M Task Builder will search the global symbol table of library TESTLIB.OLB to resolve the symbols.

3. **$ MCR FORTRAN SCANLINE=SCANLINE**
   $ LIBRARY/RSX11 TESTLIB SCANLINE

   ```
   MODULE "SCANLINE" REPLACED
   ```

   The MCR FORTRAN command compiles a source program named SCANLINE.FOR and creates the object module SCANLINE.OBJ. The LIBRARY/RSX11 command in this example uses the default function, /REPLACE, to replace the module SCANLINE in the library TESTLIB.OLB with the new version.

4.
   ```
   $ LIBRARY/RSX11/DELETE=STARTUP TESTLIB
   MODULES DELETED:

   STARTUP

   $ LIBRARY/RSX11/LIST/NAMES TESTLIB

   DIRECTORY OF FILE TESTIB.OLB;2
   OBJECT MODULE LIBRARY CREATED BY LBR:  VX129.0
   LAST INSERT OCCURRED 10-JUN-78 AT 16:41:23
   MNT ENTRIES ALLOCATED:  75 AVAILABLE:  73
   EPT ENTRIES ALLOCATED:  275;  AVAILABLE:  270
   FILE SPACE AVAILABLE:  18347 WORDS

   RECOVERABLE DELETED SPACE:  00134 WORDS

   ** MODULE:ERRMSG

     ERRMSG     ERR$ERROR    ERR$FATAL    ERR$WARNING


   ** MODULE:SCANLINE

     SCANLINE

   $ LIBRARY/RSX11/COMPRESS=(BLOCKS:50)  TESTLIB
   $ PURGE TESTLIB.OLB
   ```

   The LIBRARY/RSX11 command deletes the module STARTUP from the library TESTLIB.OLB. The next LIBRARY/RSX11 command requests a listing of the contents of the library TESTLIB.OLB. The /NAMES qualifier requests a list of each of the global symbols in the modules.

   The listing indicates that the deletion of the module STARTUP resulted in unused space; the /COMPRESS function deletes the space and requests that 50 blocks be allocated for the library. By default, /COMPRESS creates a new version of the library. The PURGE command purges the earlier version.

5.
```
$ LIBRARY/RSX11/EXTRACT=(DESCRIPTOR,RDTERM,WRTERM) -
$_/OUTPUT=TEMP -
$_DBB2:[GOODWIN.LIB]LOCALMAC.MLB
```

The /EXTRACT qualifier names three macros, DESCRIPTOR, RDTERM, and WRTERM, to be copied from the macro library LOCALMAC.MLB in the subdirectory [GOODWIN.LIB] on the disk DBB2. The /OUTPUT qualifier requests that the output file have a file name of TEMP. The LIBRARY/RSX11 command writes the output to the file TEMP.MAC in your current default disk and directory.

6.
```
$ LIBRARY/RSX11/MACRO/CREATE=(BLOCKS:40,MODULES:100) -
$_MYMAC TEMP
$ MACRO MYMAC/LIBRARY+CYGNUS/OBJECT
```

The LIBRARY/RSX11 command creates a macro library named MYMAC.MLB from the macros in the file TEMP.MAC. The new library has room for 100 modules in a 40-block file. If the input file contains multiple macros, each macro is entered in the new library.

The MACRO command assembles the source file CYGNUS.MAC; the /LIBRARY qualifier specifies the library MYMAC.MLB as an input file. If the source file CYGNUS contains any .MCALL assembly directives referencing macros not defined within the file, the assembler searches the library.

7.
```
$ LIBRARY/RSX11/LIST=MYMAC.LIS/FULL MYMAC.MLB
```

The LIBRARY/RSX11 command requests a full listing of the macro library MYMAC.MLB; the output is written to a file named MYMAC.LST.

LINK

Invokes the VAX-11 Linker to link one or more object modules into a program image and defines execution characteristics of the image. This command is described in detail in the VAX-11 Linker Reference Manual.

Format

```
LINK    file-spec[,...]

Command Qualifiers                     Defaults

/BRIEF
/[NO]CONTIGUOUS                        /NOCONTIGUOUS
/[NO]CROSS_REFERENCE                   /NOCROSS_REFERENCE
/[NO]DEBUG[=file-spec]                 /NODEBUG
/[NO]EXECUTABLE[=file-spec]            /EXECUTABLE
/FULL
/HEADER
/[NO]MAP[=file-spec]                   /NOMAP
/POIMAGE
/PROTECT
/[NO]SHAREABLE[=file-spec]             /NOSHAREABLE
/[NO]SYMBOL_TABLE[=file-spec]          /NOSYMBOL_TABLE
/[NO]SYSLIB                            /SYSLIB
/[NO]SYSSHR                            /SYSSHR
/[NO]SYSTEM[=base-address]             /NOSYSTEM
/[NO]TRACEBACK                         /TRACEBACK
/[NO]USERLIBRARY[=(table[,...])]

File Qualifiers                        Defaults

/INCLUDE=(module-name[,...])           None.
/LIBRARY
/OPTIONS
/SELECTIVE_SEARCH
```

Prompts

File: file-spec[,...]

Command Parameters

file-spec[,...]

Specifies one or more input files. The input files can be object modules to be linked, 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. If you specify multiple input files, separate the file specifications with commas (,) or plus signs (+). In either case, the linker creates a single image file.

If you do not specify a file type in an input file specification, the linker supplies default file types, based on the nature of the file. All object modules are assumed to have file types of OBJ.

No wild card characters are allowed in the file specification.

Command Qualifiers

/BRIEF

Requests the linker to produce a brief map (memory allocation) file. /BRIEF is valid only if /MAP is also specified.

A brief form of the map contains:

- 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

Controls whether the output image file is contiguous. By default, the image file is not contiguous.

/CROSS_REFERENCE
/NOCROSS_REFERENCE

Controls whether the memory allocation listing (map) contains a symbol cross reference. /CROSS_REFERENCE is valid only if /MAP is also specified and /BRIEF is not specified.

A symbol cross reference lists 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 and/or traceback information for the debugger, you can specify /DEBUG to include the information in the image as well. If the object module does not contain symbol table and/or traceback information, and you specify /DEBUG, only global symbols are available for symbolic debugging.

The /DEBUG qualifier optionally accepts the name of an alternate, user-specified debugger. If a file specification is entered, and it does not contain a file type, the linker assumes the default file type of OBJ. If you specify /DEBUG without a file specification, the default VAX/VMS Symbolic Debugger is linked with the image. For information on using the debugger, see the VAX-11 Symbolic Debugger Reference Manual.

No wild card characters are allowed in the file specification.

/EXECUTABLE[=file-spec]
/NOEXECUTABLE

Controls whether the linker creates an executable image and optionally provides a file specification for the output image file.

By default, the linker creates an executable image with the same file name as the first input file and a file type of EXE. When you specify /EXECUTABLE, you can control the defaults applied to the output file specification by the placement of the qualifier in the command, as described in Section 5.3.3, "Rules for Entering Output File Qualifiers."

You can use /NOEXECUTABLE to test a set of qualifiers, options, or input object modules, without creating an image file.

No wild card characters are allowed in the file specification.

/FULL

Requests the linker to produce a full map (memory allocation) listing. /FULL is valid only if /MAP is specified.

A full listing contains the following information:

- All the information included in the brief listing
- 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 header on a system image when used in conjunction with the /SYSTEM qualifier. All other images always have headers. However, by default, system images do not have headers.

/MAP[=file-spec]
/NOMAP

Controls whether a memory allocation listing (map) is produced and optionally defines the file specification. If you specify /MAP, you can also specify /BRIEF, /FULL or /CROSS_REFERENCE to control the contents of the map. If you do not specify any of these qualifiers, the map contains:

- All the information contained in a brief listing
- 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 Section 5.3.3, "Rules for Entering Output File Qualifiers." In interactive mode, the default is /NOMAP. However, in batch mode, the default is /MAP.

/P0IMAGE

Directs the linker to create an image that is stored only in P0 address space. The linker places the stack and RMS buffers that usually go in P1 address space in P0 address space. The /P0IMAGE qualifier is used to create executable images that modify P1 address space. See the VAX-11 Architecture Handbook for a description of P0 and P1 address space.

/PROTECT

When used in conjunction with the /SHAREABLE qualifier, the /PROTECT qualifier directs the linker to create a protected shareable image. A protected shareable image can execute privileged change mode instructions even when it is linked into a nonprivileged executable image.

/SHAREABLE[=file-spec]
/NOSHAREABLE

Requests the linker to produce a shareable image file rather than an executable image.

Shareable images cannot be run with the RUN command. However, they can be linked with object modules to create executable images. By default, the linker creates an executable image. If you specify both of the qualifiers /EXECUTABLE and /SHAREABLE, the /SHAREABLE qualifier always 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, as described in Section 5.3.3, "Rules for Entering Output File Qualifiers."

To specify an input shareable image, the /SHAREABLE qualifier must be used as an input file qualifier in an options file. See the VAX-11 Linker Reference Manual.

No wild card characters are allowed in the file specification.

/SYMBOL_TABLE[=file-spec]
/NOSYMBOL_TABLE

Requests the linker to create a separate file containing symbol definitions for all global symbols in the image. The output file will be in object module format.

If you also specify /DEBUG, the linker includes the global symbol definitions in the image for use by the debugger, and also creates a separate symbol table file.

The symbol table file can be used as input to subsequent LINK commands, to provide the symbol definitions to other images.

By default, the linker does not create a symbol table file.

When you specify /SYMBOL_TABLE, you can control the defaults applied to the output file specification, as described in Section 5.3.3, "Rules for Entering Output File Qualifiers." The output file type defaults to STB.

No wild card characters are allowed in the file specification.

/SYSLIB
/NOSYSLIB

Controls whether the default system library is to be automatically searched for unresolved references. The default system library consists of the shareable image VMSRTL.EXE and the module library STARLET.OLB.

By default, the linker searches the shareable image SYS$LIBRARY:VMSRTL.EXE and then the library SYS$LIBRARY:STARLET.OLB when it cannot resolve references using the input file(s) specified in the command.

If you specify /NOSYSLIB, neither VMSRTL.EXE nor STARLET.OLB is searched.

/SYSSHR
/NOSYSSHR

Controls whether the linker searches the default system shareable image VMSRTL.EXE when it cannot resolve references in the input file(s) specified.

By default, the linker searches the shareable image VMSRTL.EXE and then the object module library STARLET.OLB when it cannot resolve references using the input file(s) specified. Use the /NOSYSSHR qualifier to request that only STARLET.OLB be searched.

/SYSTEM[=base-address]
/NOSYSTEM

Requests the linker to produce a system image and optionally defines a base address for the image. A system image cannot be run with the RUN command; it must be bootstrapped or otherwise loaded into memory.

The base address specifies the virtual memory location at which the image is to be loaded. The address can be expressed in decimal, hexadecimal, or octal format, using the radix specifiers %D, %X, or %O, respectively. If you do not specify a base address, the linker uses the default address of %X80000000.

If you specify /SYSTEM, you cannot specify either /SHAREABLE or /DEBUG.

System images are intended for special purposes, such as standalone 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
/NOTRACEBACK

Controls whether the linker includes traceback information in the image file. By default, the linker includes traceback information so that the system can trace the call stack when an error occurs. If you specify /NOTRACEBACK, there is no traceback reporting when an error occurs.

If you specify /DEBUG, /TRACEBACK is assumed.

/USERLIBRARY[=(table[,...])]
/NOUSERLIBRARY

Controls whether the linker searches any user-defined default libraries after it has searched any specified user libraries. When you specify the /USERLIBRARY qualifier, the linker searches the process, group and system logical name tables to find the file specifications of the user-defined libraries. (The VAX-11 Linker Reference Manual explains user-defined default libraries.) You can specify the following tables for the linker to search:

ALL	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 specify neither /NOUSERLIBRARY nor /USERLIBRARY=(table), the linker assumes /USERLIBRARY=ALL by default.

The /NOUSERLIBRARY qualifier tells the linker not to search any user-defined default libraries.

File Qualifiers

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

Indicates that the associated input file is an object module library, and that only the module names specified are to be unconditionally included as input to the linker.

At least one module name must be specified. If you specify more than one module name, separate them with commas and enclose the list in parentheses.

If you specify /INCLUDE, you can also specify /LIBRARY; then, the library is subsequently searched for unresolved references.

No wild card characters are allowed in the module name specification(s).

/LIBRARY

Indicates that the associated input file is a library to be searched for modules to resolve any undefined symbols in the input files.

If the associated input file specification does not include a file type, the linker assumes the default file type of OLB. You cannot 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.

/OPTIONS

Indicates that the associated input file contains a list of options to control the linking. If you specify /OPTIONS and the associated input file specification does not include a file type, the linker uses the default file type of OPT.

For complete details on the contents of an options file, see the VAX-11 Linker Reference Manual.

/SELECTIVE_SEARCH

Indicates that the associated input file is an object module, and that any symbols defined in the module that are not necessary to resolve outstanding references should be excluded from the symbol table of the output image file and also from the symbol table file, if /SYMBOL_TABLE is specified. The binary code in the object module is always included.

Examples

1.
```
$ LINK ORION
```

   The linker 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 linker 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 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-11 DEBUG V2.0

%DEBUG-I-INITIAL, language is FORTRAN, module set to
'WEATHER'
DBG>
```

   The linker 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.

LINK/RSX11

Invokes the RSX-11M Task Builder to build an RSX-11M image. The /RSX11 qualifier is required.

For more information on the RSX-11M Task Builder, see the RSX-11M/M-PLUS Task Builder Manual.

Format

```
LINK/RSX11    file-spec[,...]


    Additional
Command Qualifiers                Defaults

/BRIEF                            /BRIEF
/[NO]DEBUG[=file-spec]            /NODEBUG
/DEFAULT_LIBRARY=file-spec
/[NO]EXECUTABLE[=file-spec]       /EXECUTABLE
/[NO]EXIT[=n]                     /NOEXIT
/FULL                             /BRIEF
/[NO]HEADER                       /HEADER
/[NO]MAP[=file-spec]              /NOMAP
/[NO]OVERLAY_DESCRIPTION          /NOOVERLAY_DESCRIPTION
/[NO]POSITION_INDEPENDENT         /NOPOSITION_INDEPENDENT
/[NO]POST_MORTEM                  /NOPOST_MORTEM
/[NO]SEQUENTIAL                   /NOSEQUENTIAL
/[NO]SYMBOL_TABLE                 /NOSYMBOL_TABLE
/TKB_OPTIONS=file-spec
/[NO]TRACE                        /NOTRACE


File Qualifiers                   Defaults

[NO]CONCATENATED                  /CONCATENATED
/INCLUDE=(module[,...])
/LIBRARY
/SELECTIVE_SEARCH
```

Prompts

File: file-spec[,...]

Command Parameters

file-spec[,...]

File specifications of one or more input files. The input files may be object modules to be linked or libraries to be searched for external references. If multiple input files are specified, they may be separated either with commas (,) or plus signs (+). In either case, a single RSX-11M image file is produced.

If a file specification does not contain a file type, the task builder supplies default file types, based on the nature of the file. All object modules are assumed to have a file type of OBJ; libraries are assumed to have a file type of OLB; overlay descriptor files are assumed to have a file type of ODL.

No wild card characters are allowed in the file specification.

Additional Command Qualifiers

/BRIEF

Requests the task builder to produce a brief map (memory allocation) file; /BRIEF is the default if /MAP is specified. The /BRIEF qualifier is valid only if /MAP is also specified. A brief form of the map contains:

- A summary of the image attributes
- A list of all segments in the image
- A summary of task builder statistics

/DEBUG[=file-spec]
/NODEBUG

Controls whether or not an image is bound with a debugger. The /DEBUG qualifier optionally accepts the name of an alternate, user-specified debugging aid. If a file specification is entered, and it does not contain a file type, the task builder assumes the default file type of OBJ.

If you specify /DEBUG without a file specification, the default debugger, ODT, is used.

/DEFAULT_LIBRARY=file-spec

Defines an object module library to use in place of the default system library, SYSLIB.OLB. The specified library is searched after all libraries specified as input files when unresolved references are encountered.

No wild card characters are allowed in the file specification.

/EXECUTABLE[=file-spec]
/NOEXECUTABLE

Controls whether or not the task builder produces an executable image and optionally provides a file specification for the output file.

By default, the task builder creates an image with the same file name as the first input file and a file type of EXE. If the first input file you specify is the name of a library qualified with /INCLUDE, then the default file name for the object module created is the same as the name of the first, or only module specified with /INCLUDE.

Use /NOEXECUTABLE when you want to determine the outcome of task building a set of modules without incurring the task builder overhead required to create an image.

When you specify /EXECUTABLE, you can control the defaults applied to the output file specification by the placement of the qualifier in the command, as described in Section 5.3.3, "Rules for Entering Output File Qualifiers." The output file type defaults to EXE.

/EXIT[=n]
/NOEXIT

Controls whether the task builder exits after a specified number of error diagnostics. By default, the task builder does not exit because of diagnostic errors. If you specify /EXIT, the task builder exits after n diagnostic errors (n is assumed to be a decimal number, by default). If you specify /EXIT and do not specify a value for n, it defaults to a value of 1.

/FULL

Requests the task builder to produce a full map (memory allocation) listing. The /FULL qualifier is valid only if /MAP is specified. A full map contains the following information:

- All the information included in the brief map
- A file contents section for each module in the image
- A list of global symbol definitions by module
- A list of unresolved global symbol references

/HEADER
/NOHEADER

Controls whether the task builder includes a task header in the image and in the symbol table file.

/MAP[=file-spec]
/NOMAP

Controls whether or not a memory allocation listing (map) is produced and optionally defines the file specification. If /MAP is specified, the qualifiers /BRIEF or /FULL can also be specified to control the contents of the map. If neither of these qualifiers is specified, /BRIEF is the default.

When you specify /MAP, you can control the defaults applied to the output file specification by the placement of the qualifier in the command, as described in Section 5.3.3, "Rules for Entering Output File Qualifiers." The output file type defaults to MAP.

No wild card characters are allowed in the file specification.

/OVERLAY_DESCRIPTION
/NOOVERLAY_DESCRIPTION

Indicates whether the input file describes an overlay structure for the image. If the input file specification does not contain a file type, the task builder uses the default file type of ODL. If you specify /OVERLAY_DESCRIPTION, you can specify only a single input file; the input file must contain the input file specifications and an overlay description.

/POSITION_INDEPENDENT
/NOPOSITION_INDEPENDENT

Indicates whether or not the image being built contains position independent code. By default, the task builder assumes that code is not position independent.

/POST_MORTEM
/NOPOST_MORTEM

Controls whether the task builder sets the Post Mortem Dump flag. If you specify /POST_MORTEM, the system automatically lists the contents of memory when the image terminates abnormally.

/SEQUENTIAL
/NOSEQUENTIAL

Controls whether the task builder reorders program sections alphabetically when it creates the image. If you specify /SEQUENTIAL, the task builder orders program sections in the order in which they are input.

/SYMBOL_TABLE[=file-spec]
/NOSYMBOL_TABLE

Requests the task builder to create a separate file in object module format containing symbol definitions for all symbols contained in the image.

If /DEBUG is specified, the task builder includes the symbol definitions in the image for use by the debugger, and also creates a separate symbol table file.

The symbol table file can be used as input to subsequent LINK/RSX11 commands, to provide the symbol definitions to other images.

By default, the task builder does not create a symbol table file. When you specify /SYMBOL_TABLE, you can control the defaults applied to the output file specification by the placement of the qualifier in the command, as described in Section 5.3.3, "Rules for Entering Output File Qualifiers." The output file type defaults to STB.

No wild card characters are allowed in the file specification.

/TKB_OPTIONS=file-spec

Specifies the name of a file containing task builder options. If the file specification does not include a file type, the default file type of CMD is assumed. You must omit the initial slash character (/) in an options file specified as input to the LINK/RSX11 command.

No wild card characters are allowed in the file specification.

/TRACE
/NOTRACE

Indicates whether the image is traceable. If you specify /TRACE, a trace trap occurs following the execution of each instruction when the image is executed.

File Qualifiers

/CONCATENATED
/NOCONCATENATED

Indicates whether the associated input file consists of concatenated object modules. By default, the task builder includes in the image all the modules in the file. If you specify /NOCONCATENATED, the task builder includes only the first module in the file.

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

Indicates that the associated input file is an object module library, and that only the module names specified are to be unconditionally included as input to the task builder.

At least one module name must be specified. If you specify more than one module name, separate them with commas and enclose the list in parentheses.

If you specify /INCLUDE, you cannot specify /LIBRARY; if you want the library to also be searched for unresolved references, you must specify the library file specification a second time.

/LIBRARY

Indicates that the associated input file is an RSX-11M object module library that is to be searched for modules that resolve any undefined symbols in the input files.

If the associated input file specification does not include a file type, the task builder assumes the default file type of OLB.

You cannot specify a library as the first input file.

/SELECTIVE_SEARCH

Indicates that the associated input file is an object module, and that any symbols defined in the module that are not necessary to resolve outstanding references should be excluded from the symbol table of the output image file and also from the symbol table file, if /SYMBOL_TABLE is specified. The binary code in the object module is always included.

Examples

1.
```
$ LINK/RSX11 AVERAGE
$ RUN AVERAGE
```

   The object module AVERAGE.OBJ is linked to create the task image named AVERAGE.EXE. The RUN command executes the task.

2.
```
$ LINK/RSX11 WEATHER,MATHLIB -
$_/INCLUDE=(TEMP,PRECIP), -
$_MATHLIB/LIBRARY -
$_/DEBUG/EXECUTABLE=DBGWEATH
```

   The task builder links the object module WEATHER.OBJ with the modules TEMP and PRECIP from the library MATHLIB.OLB; MATHLIB is respecified as an input file with the /LIBRARY qualifier to indicate that MATHLIB should also be searched for any unresolved references.

   The image includes the system debugging aid, ODT. The output RSX-11M image file is named DBGWEATH.EXE.

3.
```
$ COBOL/RSX11 PAYROLL
$ RUN SYS$SYSTEM:MRG
PLEASE ENTER FILE SPECIFICATION FOR OUTPUT FILE
PAYROL.ODL
    .
    .
    .

$ LINK/RSX11 PAYROL/OVERLAY
```

   The COBOL/RSX11 command invokes the PDP-11 COBOL-74/VAX compiler to compile a source program named PAYROLL.CBL. The LINK/RSX11 command specifies the name of the overlay description file, PAYROL.ODL, created by the MERGE program.

Login Procedure

There is no LOGIN command. Rather, you get the attention of the system and signal your intention to access the system by pressing CTRL/C, CTRL/Y, or carriage return on a terminal not currently in use. The system then prompts for your user name and your password, and validates them.

Specify the optional qualifiers immediately after you enter your user name.

Format

```
    CTRL/C    or   CTRL/Y   or  RET


     Qualifiers                         Defaults

     /CLI=command-interpreter           None.
     /DISK=device-name[:]
```

Prompts

```
Username:  user-name[qualifier,...]

Password:  password
```

Command Parameters

None.

Description

The login procedure:

- Validates your right to access the system by checking your user name and password against the entries in the user authorization file.
- Establishes the default characteristics of your terminal session based on your user name entry in the authorization file.
- 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.

Command Qualifiers

/CLI=command-interpreter

Specifies the name of an alternate command interpreter to override the default command interpreter listed in the user authorization file. The command interpreter you designate must be in SYS$SYSTEM and named command-interpreter.EXE.

/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.

Examples

1.
```
CTRL/Y
Username:  HUMPTY
Password:
```

CTRL/Y gets the attention of the operating system, which immediately prompts for user name. After validating the user name, the system prompts for the password, but does not echo it.

2.
```
RET
Username:  HIGGINS/DISK=DBB2
Password:
        Welcome to VAX/VMS Version 2.00

$ SHOW DEFAULT
  DBB2:[HIGGINS]
```

The /DISK qualifier requests that the default disk for the terminal session be DBB2. The SHOW DEFAULT command response shows that DBB2 is the default disk.

3.
```
CTRL/C

Username:  LIZA/CLI=MCR
Password:
        Welcome to VAX/VMS Version 2.00

>
```

The /CLI qualifier requests the alternate MCR command interpreter. The right angle bracket (>) indicates that MCR is active and expects an MCR command.

LOGOUT

Terminates an interactive terminal session.

Format

```
LOGOUT

Command Qualifiers        Defaults

/BRIEF                    None.
/FULL
```

Prompts

None.

Description

You must use the LOGOUT command to end a terminal session. If you turn the power off at your terminal without using the LOGOUT command from a direct line, you remain logged in.

Remember that if you have been using the SET HOST command to log in on remote processors, multiple LOGOUT commands may be required to end the terminal session.

Command Qualifiers

/BRIEF

Requests the brief form of the logout message. The command interpreter displays your user name and the date and time at which you logged out. The default for an interactive job is /BRIEF.

/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.

Examples

1.
```
$ LOGOUT
  HIGGINS   logged out at 23-JAN-1979 17:48:56.73
```

2.
```
$ LOGOUT/FULL
  HIGGINS      logged out at 24-JAN-1979 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
  Elapsed CPU time: 0 00:01:30.50  Elapsed time:     0 04:59:02.63
```

The LOGOUT command with the /FULL qualifier displays a summary of accounting statistics for the terminal session.

MACRO

Invokes the VAX-11 MACRO assembler to assemble one or more assembly language source programs.

For more information on the VAX-11 MACRO assembler, see the VAX-11 MACRO User's Guide.

Format

```
MACRO   file-spec[,...]

Command Qualifiers                          Defaults

None.                                       None.

File Qualifiers                             Defaults

/[NO]CROSS_REFERENCE[=function[,...]]       /NOCROSS_REFERENCE
/DISABLE=(function[,...])                   /DISABLE=(ABSOLUTE,DEBUG,-
                                               TRUNCATION,SUPPRESSION)
/ENABLE=(function[,...])                    /ENABLE=(GLOBAL,TRACEBACK)
/LIBRARY
/[NO]LIST[=file-spec]                       (see text)
/[NO]OBJECT[=file-spec]                     (see text)
/[NO]SHOW[=(function[,...])]                /SHOW=(CONDITIONAL,CALLS,-
                                               DEFINITIONS)
/UPDATE[=(update-file-spec[,...])]
```

Prompts

File: file-spec[,...]

Command Parameters

file-spec[,...]

Specifies one or more VAX-11 MACRO assembly language source files to be assembled. If you do not specify a file type for an input file, the assembler uses the default file type of MAR.

You can specify more than one input file. If you separate the file specifications with commas (,), each file is assembled separately. If you separate the file specifications with plus signs (+), the files are concatenated and assembled as a single input file, producing single object and listing files.

No wild card characters are allowed in the file specification(s).

File Qualifiers

/CROSS_REFERENCE[=function[,...]]
/NOCROSS_REFERENCE[=function[,...]]

Controls whether a cross reference listing is included in the listing file. The /CROSS_REFERENCE qualifier includes a cross reference listing, and therefore requires that a listing file exist. The /NOCROSS_REFERENCE qualifier excludes it. You can specify one or more of the functions listed below.

If you specify /CROSS_REFERENCE without any functions, it is equivalent to /CROSS_REFERENCE=(MACROS,SYMBOLS).

ALL	Cross reference directives, macros, operation codes, registers, and symbols
DIRECTIVES	Cross reference directives
MACROS	Cross reference macros
OPCODES	Cross reference operation codes
REGISTERS	Cross reference registers
SYMBOLS	Cross reference symbols

If you specify more than one function, separate them by commas and enclose the list in parentheses.

/DISABLE=(function[,...])
/ENABLE=(function[,...]

Provides initial settings for the functions controlled by the assembler directives .ENABLE and .DISABLE. You must specify at least one of the functions listed below. You can enable or disable:

ABSOLUTE	Assembly of relative addresses as absolute addresses
DEBUG	Inclusion of local symbol table information in the object file for use with the debugger
TRUNCATION	Truncation of floating-point numbers (if truncation is disabled, numbers are rounded)
GLOBAL	Assumption that undefined symbols in the assembly are external symbols
SUPPRESSION	Suppression of the listing of unreferenced symbols in the symbol table
TRACEBACK	Providing information to the debugger traceback mechanism

The default is /ENABLE=(GLOBAL,TRACEBACK).

If you specify more than one function, separate them by commas and enclose the list in parentheses.

/LIBRARY

Indicates that the associated input file is a macro library. If you do not specify a file type, the assembler uses the default file type of MLB.

If you specify more than one macro library as input files, the libraries are searched in reverse order of their specification when a macro call is issued in a source program.

You must not specify the /LIBRARY and /UPDATE qualifiers at the same time; they are mutually exclusive.

/LIST[=file-spec]
/NOLIST

Controls whether an output listing is created, and optionally provides an output file specification for the listing file.

If you issue the MACRO command interactively, the assembler, by default, does not create a listing file. When /NOLIST is present, either explicitly or by default, errors are reported on the current output device.

If you execute the MACRO command in a batch job, /LIST is the default. When you specify /LIST, you can control the defaults applied to the output file specification by the placement of the qualifier in the command, as described in Section 5.3.3, "Rules for Entering Output File Qualifiers." The default file type provided for listing files is LIS.

No wild card characters are allowed in the file specification.

/OBJECT[=file-spec]
/NOOBJECT

Controls whether an object module is created by the assembler. It also defines the file specification for the file.

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. When you specify /OBJECT, you can control the defaults applied to the output file specification by the placement of the qualifier in the command, as described in Section 5.3.3, "Rules for Entering Output File Qualifiers."

No wild card characters are allowed in the file specification.

/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 functions listed below. If you specify /SHOW without any functions, the listing level count is incremented. If you specify /NOSHOW without any functions, the listing level count is decremented.

The /SHOW qualifier requires that a listing file exist.

CONDITIONALS	List unsatisfied conditional code associated with .IF and .ENDC directives
CALLS	List macro calls and repeat range expansions
DEFINITIONS	List macro definitions
EXPANSIONS	List macro expansions
BINARY	List binary code generated by the expansion of macro calls.

If you omit the /SHOW qualifier, the default is equivalent to /SHOW=(CONDITIONALS,CALLS,DEFINITIONS).

If you specify more than one function, separate them by commas and enclose the list in parentheses.

For additional details on these functions, see the VAX-11 MACRO Language Reference Manual.

/UPDATE[=(update-file-spec[,...])]

Indicates that the associated input file is to be updated with the specified update file(s). The batch editor SLP is used. Updating is described in the VAX-11 MACRO User's Guide. The batch editor, SLP, is described in the VAX-11 Utilities Reference Manual.

By default, the assembler uses update files with the same name as the input source file and a file type of UPD.

When multiple update files are specified with the /UPDATE qualifier, the assembler merges the contents into a single list of updates before applying the updates to the source file. You must separate multiple update files with commas and enclose the list in parentheses.

As a result of the update, the input source file and update file(s) remain unchanged. The effects appear in the compiled output. The listing also provides an audit trail of the changes if you specify /LIST with /UPDATE.

No wild card characters are allowed in the update file specifications.

You must not specify the /LIBRARY and /UPDATE qualifiers at the same time; they are mutually exclusive.

If an update file is not found, the assembler prints an informational message but continues the assembly.

Examples

1.
```
$ MACRO  ORION
```

   The MACRO assembler assembles the file ORION.MAR and creates an object file named ORION.OBJ. If this command is executed in a batch job, the assembler also creates a listing file named ORION.LIS.

2.
```
$ MACRO/LIST  CYGNUS, LYRA/OBJECT=LYRAN
```

   This MACRO command requests two separate assemblies. The MACRO command 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.

3.
```
$ MACRO  ALPHA/LIST+MYLIB/LIBRARY-
$_ + [TEST]OLDLIB/LIBRARY + []BETA
$ PRINT ALPHA
```

   The MACRO command concatenates the files ALPHA.MAR and BETA.MAR to produce an object file named ALPHA.OBJ and a listing file named ALPHA.LIS. MYLIB.MLB (in the current default directory) and OLDLIB.MLB (in the directory TEST) are specified as libraries to be searched for macro definitions. When macro calls are found in BETA.MAR, OLDLIB, MYLIB, and the system library STARLET.MLB are searched, in that order, for the definitions.

   The PRINT command prints the listing file ALPHA.LIS.

Invokes the MACRO-11 assembler to assemble one or more MACRO-11 assembly language source programs.

For more information on the MACRO-11 assembler, see the IAS/RSX-11 MACRO-11 Reference Manual.

Format

```
MACRO/RSX11  file-spec[,...]


    Additional
Command Qualifiers                    Defaults

None.                                 None.


File Qualifiers                       Defaults

/DISABLE=(function[,...])             /DISABLE=(ABSOLUTE,DEBUG,-
                                         TRUNCATION,SUPPRESSION)
/LIBRARY
/[NO]LIST[=file-spec]                 (see text)
/[NO]OBJECT[=file-spec]               (see text)
/[NO]SHOW[=(function[,...])]          /SHOW=(CONDITIONALS,CALLS,-
                                         DEFINITIONS)
```

Prompts

File: file-spec[,...]

Command Parameters

file-spec[,...]

Specifies one or more MACRO-11 assembly language source files to be assembled. If you do not specify a file type for an input file, the assembler uses the default file type of MAC.

You can specify more than one input file. If you separate the file specifications with commas (,), each file is assembled separately. If you separate the file specifications with plus signs (+), the files are concatenated and assembled as a single input file, producing single object and listing files.

File Qualifiers

/DISABLE=(function[,...])

Provides initial settings for the functions controlled by the assembler directives .ENABLE and .DISABLE. You must specify at least one of the functions listed below. You can disable:

ABSOLUTE	Assembly of relative addresses as absolute addresses
DEBUG	Inclusion of local symbol table information in the object file for use with the debugger
TRUNCATION	Truncation of floating-point numbers (if truncation is disabled, numbers are rounded)
GLOBAL	Assumption that undefined symbols in the assembly are external symbols
SUPPRESSION	Suppression of listing of unreferenced symbols in the symbol table
TRACEBACK	Providing information to the debugger traceback mechanism

If you specify more than one function, separate them by commas and enclose the list in parentheses.

/LIBRARY

Indicates that the associated input file is a macro library. If you do not specify a file type, the assembler uses the default file type of MLB.

If you specify more than one macro library as input files, the libraries are searched in reverse order of their specification when a macro call is issued in a source program.

/LIST[=file-spec]
/NOLIST

Controls whether an output listing is created, and optionally provides an output file specification for the listing file.

If you issue the MACRO command interactively, the assembler, by default, does not create a listing file. When /NOLIST is present, either explicitly or by default, errors are reported on the current output device.

If you execute the MACRO command in a batch job, /LIST is the default. When you specify /LIST, you can control the defaults applied to the output file specification by the placement of the qualifier in the command, as described in Section 5.3.3, "Rules for Entering Output File Qualifiers." The default file type provided for listing files is LST.

No wild card characters are allowed in the file specification.

/OBJECT[=file-spec]
/NOOBJECT

Controls whether an object module is created by the assembler. It also defines the file specification for the file.

By default, the assembler creates an object module with the same file name as the first input file and a file type of OBJ. When you specify /OBJECT, you can control the defaults applied to the output file specification by the placement of the qualifier in the command, as described in Section 5.3.3, "Rules for Entering Output File Qualifiers." The default file type provided for object files is OBJ.

No wild card characters are allowed in the file specification.

/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 functions listed below. If you specify /SHOW without any functions, the listing level count is incremented. If you specify /NOSHOW without any functions, the listing level count is decremented.

The /SHOW qualifier requires that a listing file exist.

CONDITIONALS	List unsatisfied conditional code associated with .IF and .ENDC directives
CALLS	List macro calls and repeat range expansions
DEFINITIONS	List macro definitions
EXPANSIONS	List macro expansions
BINARY	List binary code generated by the expansion of macro calls.

If you omit the /SHOW qualifier, the default is equivalent to /SHOW=(CONDITIONALS,CALLS,DEFINITIONS).

If you specify more than one function, separate them by commas and enclose the list in parentheses.

For additional details on these functions, see the IAS/RSX-11 MACRO-11 Reference Manual.

Examples

1. `$ MACRO/RSX11  ORION`

   The MACRO-11 assembler assembles the file ORION.MAC and creates an object file named ORION.OBJ. If this command is executed in a batch job, the assembler also creates a listing file named ORION.LST.

2.
```
$ MACRO/RSX11/LIST  CYGNUS, LYRA/OBJECT=LYRAN
```

   This MACRO/RSX11 command requests two separate assemblies. The MACRO-11 assembler assembles CYGNUS.MAC to produce CYGNUS.LST and CYGNUS.OBJ. Then it assembles LYRA.MAC and creates a listing file named LYRA.LST and an object module named LYRAN.OBJ.

3.
```
$ MACRO/RSX11  ALPHA/LIST+MYLIB/LIBRARY-
$_ + [TEST]OLDLIB/LIBRARY + []BETA
$ PRINT ALPHA
```

   The MACRO/RSX11 command concatenates the files ALPHA.MAC and BETA.MAC to produce an object file named ALPHA.OBJ and a listing file named ALPHA.LST. MYLIB.MLB (in the current default directory) and OLDLIB.MLB (in the directory TEST) are specified as libraries to be searched for macro definitions. When macro calls are found in BETA.MAC, OLDLIB, MYLIB, and the system library STARLET.MLB are searched, in that order, for the definitions.

   The PRINT command prints the listing file, ALPHA.LST.

Invokes the VAX/VMS Personal Mail Utility. This utility is used to send messages to other users of the computer system.

The MAIL utility is described in the VAX-11 Utilities Reference Manual.

Format

```
    MAIL  [file-spec]  [username[,...]]


    Command Qualifiers                          Defaults

    /SUBJECT="text"                             None.
```

Prompts

Username: username[,...]

Command Parameters

file-spec

Specifies an optional file containing message text to be sent to the specified user(s). If you omit the file type, the default file type is TXT. The username parameter is required with the file-spec parameter.

If you omit the file-spec parameter, the MAIL utility is invoked to process MAIL commands interactively.

No wild card characters are allowed in the file specification.

username[,...]

Specifies one or more users to receive the message. If the file-spec parameter is specified, this parameter is required. A user name is the name that the user uses to login. If any user is on a remote node, you should precede that username parameter with the name of the remote node followed by two colons (::).

As an alternative to listing the user names, you can specify a distribution list file containing user names. Simply precede the distribution list file specification with an at sign (@) and enclose this construction in quotation marks ("). The file you specify should contain the user names, entered one per line, denoting any remote nodes as described above. If you omit the file type, the default file type is DIS. No wild card characters are allowed in the distribution list file specification.

See the MAIL utility description in the VAX-11 Utilities Reference Manual for further details on user name specifications.

Description

When the MAIL command is specified without parameters, the MAIL utility is invoked to process MAIL commands interactively. The utility signals it is active with the prompt MAIL>.

When the MAIL command is specified with parameters, the specified message file is sent to the specified user(s). This use of the MAIL command is the same as the MAIL utility's SEND command.

Command Qualifiers

/SUBJECT=text

Specifies the subject of the message for the heading. If the text consists of more than one word, enclose the text in quotation marks (").

If you omit this qualifier, the message is sent without a subject notation.

Examples

1.
```
$ MAIL

MAIL>
```

The MAIL command has been specified without parameters, so the MAIL utility is invoked to process commands interactively.

2.
```
$ MAIL/SUBJECT="New Project" PROJECT.DOC JONES,SMITH,ADAMS
```

This MAIL command specifies that the file named PROJECT.DOC is to be sent to users JONES, SMITH, and ADAMS, with a subject description of New Project in the heading.

3.
```
$ MAIL/SUBJECT="Vacation Policy Change" NEWSLETTR "@USERS"
```

This MAIL command invokes the MAIL utility to send the file NEWSLETTR.TXT to all the users named in the file USERS.DIS. The subject description is Vacation Policy Change.

Provides a means of running RSX-11M components in a manner that is compatible with the RSX-11M operating system.

Format

```
MCR   [component [command-string]]
```

Command Qualifiers	Defaults
None.	None.

Prompts

MCR>

Command Parameters

component

The RSX-11M command used to invoke the desired component, for example, MAC or TKB. If you do not specify a component, MCR prompts for a command.

The component name must be equivalent to the file name portion of the file specification for the component as it exists on the VAX/VMS system disk, that is, SYS$SYSTEM:filename. For example, you must type MAC, not MACRO, to invoke the MACRO-11 assembler.

command-string

A valid command string for the component. If you do not specify a command string, the requested component prompts for a command string.

Description

This command does not provide all the capabilities of the MCR command interpreter (such as indirect command procedures). It can only invoke the specified image.

For information on how to use the MCR command interpreter under VAX/VMS and for a list of the RSX-11M components available under VAX/VMS, see the VAX-11/RSX-11M User's Guide.

Examples

1.
```
$ MCR DSP MYFILE.DAT
$
```

   The MCR command precedes a single RSX-11M command. When the command finishes, DCL prompts for another command.

2.
```
$ MCR
MCR>PIP MYFILE.DAT/SP
MCR>^Z
$
```

   The MCR command requests activation of MCR command mode. The MCR> prompt indicates that the MCR command interpreter is ready to accept commands. After the PIP command executes, MCR continues prompting until you use CTRL/Z to return to DCL.

3.
```
$ MCR PIP
PIP> MYFILE.DAT/SP
PIP> ^Y
```

   The MCR command requests PIP, which in turn prompts for a command string. Pressing CTRL/Y returns control to DCL.

MESSAGE

Invokes the VAX-11 Message Utility to compile one or more files of message definitions. The Message Utility is described in detail in the VAX-11 Utilities Reference Manual.

Format

```
MESSAGE    file-spec[,...]

Command Qualifiers            Defaults

/[NO]FILE_NAME[=file-spec]    /NOFILE_NAME
/[NO]LIST[=file-spec]         (see text)
/[NO]OBJECT[=file-spec]       /OBJECT
/[NO]SYMBOLS                  /SYMBOLS
/[NO]TEXT                     /TEXT
```

Prompts

File: file-spec[,...]

Command Parameters

file-spec[,...]

Specifies one or more message files to be compiled. If you do not specify a file type for an input file, the MESSAGE command uses the default file type of MSG.

You can specify more than one input file. If you separate the file specifications with either commas (,) or plus signs (+), the files are concatenated and compiled as a single input file, producing single object and listing files. If you specify SYS$INPUT as the file-spec parameter, the Message Utility input file(s) must follow the command in the input stream. In this case, both the object module file (given by the /OBJECT qualifier) and the listing file (given by the /LIST qualifier) must be explicitly named.

Wild card characters are allowed in the file specification(s). See Section 2.1.6.

Command Qualifiers

/FILE_NAME=file-spec
/NOFILE_NAME

Controls whether or not the object module contains an indirect pointer to a file of messages. The default is /NOFILE_NAME, which means that all compiled messages are in the object module. You may specify a compiled message file with /FILE_NAME=file-spec. In this case, the object module that is created contains an indirect pointer to the file you name. At execution time, the messages are sought in the file rather than in memory. Thus, you can update the message file dynamically rather than recompile or relink the entire object module. See the VAX-11 Utilities Reference Manual for further details on creating message files.

Whenever you specify /FILE_NAME, /NOTEXT is implied. The /FILE_NAME qualifier requires that the /OBJECT qualifier be explicitly or implicitly in effect.

At execution time, the default device and directory is SYS$MESSAGE and the default file type is EXE.

No wild card characters are allowed in the file specification.

/LIST[=file-spec]
/NOLIST

Controls whether an output listing is created, and optionally provides an output file specification for the listing file.

When in batch mode, the output listing is created by default. However, in interactive mode the default is to produce no output listing.

The default file type for listing files is LIS.

No wild card characters are allowed in the file specification.

/OBJECT[=file-spec]
/NOOBJECT

Controls whether an object module is created by the message compiler, and optionally provides an output file specification for the file.

By default, the compiler creates an object module with the same file name as the first input file and a file type of OBJ. The output is directed to the device and directory identified by the logical name SYS$MESSAGE.

The object module produced by default contains text with symbols, but does not contain an indirect pointer to a compiled message file. See the /SYMBOLS, /TEXT, and /FILE_NAME qualifiers for ways to alter the contents of the object module.

No wild card characters are allowed in the file specification.

/SYMBOLS
/NOSYMBOLS

Controls whether global symbols will be present in the object module specified by the /OBJECT qualifier. By default, object modules are created with global symbols. You may specify /NOSYMBOLS to eliminate global symbols from the object module. The /SYMBOLS qualifier requires that /OBJECT be explicitly or implicitly in effect.

/TEXT
/NOTEXT

Controls whether the actual message text and associated information is placed in the object module. The default is /TEXT. The /NOTEXT qualifier inhibits the creation of the data portion of the object module.

The /TEXT and /FILE_NAME qualifiers are mutually exclusive.

The /TEXT qualifier requires that the /OBJECT qualifier be explicitly or implicitly in effect.

The /NOTEXT qualifier can be used in conjunction with /SYMBOLS to produce an object module containing only global symbols.

Examples

1.
```
$ MESSAGE/LIST=WEATHER2  WEATHER
```

This MESSAGE command compiles the file WEATHER.MSG and creates the listing WEATHER2.LIS. The object module contains symbols and text and is placed in WEATHER.OBJ.

2.
```
$ MESSAGE/LIST=MSGOUTPUT/FILE_NAME=ALPHMESG  MESSAGEA
```

This MESSAGE command compiles the file MESSAGEA.MSG and creates the object module MESSAGEA.OBJ with no text, only symbols and an indirect pointer to the message file ALPHMESG. The listing file MSGOUTPUT.LIS is also created. At execution time, if no logical name exists for ALPHMESG, the text is sought from SYS$MESSAGE:ALPHMESG.EXE.

MOUNT

Makes a volume and the files or data it contains available for processing by system commands or user programs.

Format

```
MOUNT  device-name[:][,...]  [ volume-label[,...] ] [logical-name[:]]
```

Command Qualifiers[1]	Defaults
/DATA_CHECK[=option[,...]]	
/FOREIGN	
/OVERRIDE=(option[,...])	
/OWNER_UIC=uic	
/PROCESSOR=option	
/PROTECTION=code	
/[NO]WRITE	/WRITE

Qualifiers for Disks	Defaults
/ACCESSED=n	
/BIND=volume-set-name	
/[NO]CACHE[=(option[,...])]	
/EXTENSION=n	
/GROUP	
/[NO]QUOTA	/QUOTA
/[NO]SHARE	/NOSHARE
/SYSTEM	
/UNLOCK	
/WINDOWS=n	

Qualifiers for Tapes	Defaults
/BLOCKSIZE=n	/BLOCKSIZE=2048
/DENSITY=n	
/[NO]HDR3	/HDR3
/[NO]LABEL	/LABEL
/RECORDSIZE=n	

Prompts

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

Label: volume-label[,...]

Log_Name: logical-name[:]

1. For convenience, qualifiers that are applicable only to disks or to tapes are listed separately. All qualifier descriptions appear in alphabetical order, however.

Command Parameters

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

Specifies the physical device name or logical name of the device on which the volume is to be mounted.

If you specify more than one device name for a disk or tape volume set, separate the device names with either commas (,) or plus signs (+). For a tape volume set, you can specify more volume labels than device names.

volume-label[,...]

Specifies the label on the volume. For disk volumes, labels can have from 1 through 12 characters; for tape volumes, labels can have from 1 through 6 characters.

If you specify more than one volume label, separate the labels with either commas (,) or plus signs (+). The volumes must be in the same volume set and the labels must be specified in ascending order according to relative volume number.

When you mount a tape volume set, the number of volume labels need not equal the number of device names specified. When a tape reaches end-of-file, the system requests the operator to mount the next volume on one of the devices.

When you mount a disk volume set, each volume label specified in the list must correspond to a device name in the same position in the device name list.

The volume-label parameter is not required when you mount a volume with the /FOREIGN qualifier or when you specify /OVERRIDE=IDENTIFICATION. To specify a logical name when you enter either of these qualifiers, type any alphanumeric characters in the volume label parameter position.

logical-name[:]

Defines a 1- through 63-alphanumeric character string logical name to be associated with the device.

If you do not specify a logical name, the MOUNT command assigns the default logical name DISK$volume-label to individual disk devices; it assigns the default logical name DISK$volume-set-name to the device on which the root volume of a disk volume set is mounted. Similarly, if you do not specify a logical name for a tape device, the MOUNT command assigns only one logical name, TAPE$volume-label, to the first tape device in the list. No default logical volume set name is assigned.

The MOUNT command places the name in the process logical name table, unless you specify /GROUP or /SYSTEM. In the latter cases, it places the logical names in the group or system logical name tables.

Description

When you issue the MOUNT command, the MOUNT command checks:

- That the device has not been allocated to another user
- That a volume is physically loaded on the device specified
- That the label on the volume matches the label specified

Mounting volumes with the /SHARE, /GROUP, or /SYSTEM qualifiers deallocates the device.

For additional information and examples of using the MOUNT command, see Chapter 3, "Disk and Tape Volumes." For additional information on creating and using disk volume sets, see Chapter 3 of this manual and also the VAX/VMS Operator's Guide.

For more information on managing volumes using certain MOUNT command qualifiers, see the VAX/VMS System Manager's Guide.

Command Qualifiers

/ACCESSED=n

Specifies, for disk volumes, the approximate number of directories that will be in use, concurrently, on the volume. You can specify a value from 0 through 255 to override the default specified when the volume was initialized.

The user privilege OPER is required to use /ACCESSED.

/BIND=volume-set-name

Creates a volume set of one or more disk volumes or adds one or more volumes to an existing volume set.

The volume-set-name specifies a 1- through 12-alphanumeric character name identifying the volume set.

When you create a volume set, the volumes specified in the volume-label list are assigned relative volume numbers based on their position in the label list. The first volume specified becomes the root volume of the set.

When you add a volume or volumes to a volume set, the first volume label specified must be that of the root volume or the root volume must already be online.

/BLOCKSIZE=n

Specifies, for tape volumes, the default block size.

Legal values are in the range of 20 through 65532 for RMS operations and 18 through 65534 for non-RMS operations. By default, records are written to tape volumes in 2048-byte blocks. For foreign or unlabeled tapes, the default is 512-bytes.

You must specify /BLOCKSIZE when you are mounting:

- Tapes that do not have HDR2 labels. For these tapes, you must specify the block size. For example, you must specify /BLOCKSIZE=512 to mount an RT-11 tape.
- Tapes that will be written using any RSX-11M compatibility mode program except PIP (for example, editors and compilers that execute in compatibility mode). For these tapes, you must specify /BLOCKSIZE=512.
- Tapes that contain records whose size exceeds the default block size (2048 bytes). In this case, specify the size of the largest record for the block size.

/CACHE=(option[,...])
/NOCACHE

Controls whether disk caching limits established at system generation time are disabled or overridden.

The /CACHE qualifier overrides one or more of the present disk caching limits established at system generation time. You may alter one or more of the following limits through the appropriate option:

[NO]EXTENT[=n]	Enables or disables extent caching. If you enable extent caching, you must have the operator user privilege (OPER) and you must specify n, the number of entries in the extent cache. Note that NOEXTENT is equivalent to EXTENT=0; both disable extent caching.
[NO]FILE_ID[=n]	Enables or disables file identification caching. To enable file identification caching, you must have the operator user privilege (OPER) and you must specify n, the number of entries, as a value greater than one. Note that NOFILE_ID is equivalent to FILE_ID=1; both disable file identification caching.
LIMIT=n	Specifies the maximum amount of free space in the extent cache in one-thousandths of the currently available free space on the disk.
[NO]QUOTA[=n]	Enables or disables quota caching. If you enable quota caching, you must have the operator user privilege (OPER) and you must specify n, the number of entries in the quota cache. Normally n is set to the maximum number of active users expected for a disk with quotas enabled. Both NOQUOTA and QUOTA=0 disable quota file caching.
WRITETHROUGH	Disables writeback caching, which only writes the file headers of files open for write when the files are closed. Thus, if you specify the WRITETHROUGH option, file headers are written to the disk on every file header operation.

If you specify more than one option, separate them by commas and enclose the list in parentheses.

If you specify /NOCACHE, all disk caching is disabled for this volume. Note that the /NOCACHE qualifier is equivalent to /CACHE=(NOEXTENT,NOFILE_ID,NOQUOTA,WRITETHROUGH).

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

Overrides the read-check and/or write-check options specified for a volume when it was initialized. You can specify either or both of the following options:

READ	Perform checks following all read operations
WRITE	Perform checks following all write operations

If you specify more than one option, separate them by commas and enclose the list in parentheses.

If you specify /DATA_CHECK without specifying an option, the system assumes /DATA_CHECK=WRITE.

/DENSITY=n

Specifies, for foreign or unlabeled tapes, the density (in bpi) at which the tape will be written. You can specify 800, 1600, or 6250, if supported by the tape drive. If you do not specify a density for a tape that was previously written, the density defaults to that at which the tape was written.

The specified density is used only if you specify /FOREIGN or /NOLABEL and the first operation performed on the tape is a write.

If you specify /LABEL, or if the first operation on the tape is a read, the tape is read or written at the density at which the first record on the tape is recorded.

/EXTENSION=n

Specifies, for disk files, the number of blocks by which files will be extended on the volume unless otherwise specified by an individual command or program request.

You can specify a value from 0 through 65535 to override the value specified when the volume was initialized.

/FOREIGN

Indicates that the volume is not in the standard format used by the VAX/VMS operating system; that is, a tape volume is not in the standard ANSI format; or, a disk volume is not in Files-11 format.

If you mount a volume with the /FOREIGN qualifier, the program you use to read the volume must be able to process the labels on the volume, if any. The VAX/VMS operating system does not provide an ACP (ancillary control process) to process the volume.

DOS-11 and RT-11 volumes must be mounted with the /FOREIGN qualifier and processed with the File Transfer (FLX) Utility, as described in the VAX-11 Utilities Reference Manual.

The default protection applied to foreign volumes is RWLP for the system and owner. If /GROUP is also specified, group members are also given RWLP (Read, Write, Logical I/O and Physical I/O) access. If /SYSTEM or /SHARE is specified, the group and world are both given RWLP access. If you mount a volume currently in Files-11 format with the /FOREIGN qualifier, you must have the override volume protection user privilege (VOLPRO), or you must be the owner of the volume.

/GROUP

Makes the volume available to other users with the same group number in their user identification codes (UICs) as the user issuing the MOUNT command.

The logical name for the device is placed in the group logical name table. You must have the user privilege to place a name in the group logical name table (GRPNAM) to use the /GROUP qualifier.

Be aware that if the volume is owned by a group other than yours, access may be denied due to the volume protection.

/HDR3
/NOHDR3

Controls whether ANSI HDR3 labels are written on magnetic tapes. By default, HDR3 labels are written. You may specify /NOHDR3 to write tapes that will be used on other systems that do not accept HDR3 labels.

/LABEL
/NOLABEL

Indicates, for tape volumes, whether the tape contains standard labels.

If you mount a tape with the /NOLABEL qualifier, an end-of-file condition is returned when a tape mark is encountered during reading the tape. Note that /NOLABEL is equivalent to /FOREIGN.

The default protection for unlabeled tapes is all access to the system and owner, and no access to the group and world.

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

Inhibits one or more of the following protection checks that the MOUNT command performs:

ACCESSIBILITY	(For tapes only). Allows you to override a nonblank VOL1 or HDR1 accessibility field. You must have the user privilege to override volume protection (VOLPRO) or be the owner of the volume.
EXPIRATION	(For tapes only). Allows you to write a tape that has not yet reached its expiration date. You must have the user privilege to override volume protection (VOLPRO) or be the owner of the volume.

IDENTIFICATION	Allows you to mount a volume when you do not know what the volume label is. If you specify /OVERRIDE=IDENTIFICATION, you can specify anything for the volume-label parameter or you can omit it; the MOUNT command ignores whatever you enter. The volume must be mounted /NOSHARE (either explicitly or by default).
SETID	(For tapes only). Allows you to inhibit checks of the volume set identification when you switch reels in a multivolume tape set.

If you specify more than one option, separate them by commas and enclose the list in parentheses.

/OWNER_UIC=uic

Requests that the specified user identification code be assigned ownership of the volume while it is mounted, overriding the ownership recorded on the volume. Or, if you are mounting a volume /FOREIGN, requests an owner UIC other than your current UIC.

Specify the UIC in the format:

[g,m]

g	is an octal number in the range of 0 through 377 representing the group number.
m	is an octal number in the range of 0 through 377 representing the member number.

The square brackets ([ ]) are required in the UIC specification.

To use this qualifier for a Files-11 volume you must have the VOLPRO user privilege to override volume protection, or you must be the owner of the volume.

/PROCESSOR=option

Requests that the MOUNT command associate an ancillary control program (ACP) to process the volume, overriding the default manner in which ACPs are associated with devices. The specified option can be one of the following:

UNIQUE	Create a new process to execute a copy of the default ACP image for the specified device type or controller.
SAME:device	Use the same ACP process currently being used by the device specified.
file-spec	Create a new process to execute the ACP image specified by the file-spec (for example, a modified or a user-written ACP). No wild card characters are allowed in the file specification.

You must have the operator user privilege (OPER) to use the /PROCESSOR qualifier.

/PROTECTION=code

Specifies the protection code to be assigned to the volume.

Specify the code according to the standard syntax rules for specifying protection. (These rules are given in Section 5.10.) If you omit a protection category, that category of user is denied all access.

If you do not specify a protection code, the protection defaults to that assigned to the volume when it was initialized. If you specify the /PROTECTION qualifier when you mount a volume with the /SYSTEM or /GROUP qualifiers, the protection code specified overrides any access rights implied by the other qualifiers.

If you specify the /FOREIGN qualifier, the Execute and Delete access codes have no meaning. You can, however, specify the access codes P (Physical I/O) and/or L (Logical I/O) to restrict the nature of input/output operations that different user categories can perform.

To use the /PROTECTION qualifier on a Files-11 volume you must have the VOLPRO user privilege to override volume protection, or you must be the owner of the volume.

/QUOTA
/NOQUOTA

Controls whether or not disk quotas will be enforced on this disk volume. The default is /QUOTA, which enforces the quotas for each user. The /NOQUOTA qualifier inhibits this checking. You must have the VOLPRO user privilege or be the owner of the volume to specify the /QUOTA qualifier. See Section 3.4.1.3 for a discussion of disk quotas.

/RECORDSIZE=n

Specifies, for tape volumes, the number of characters in each record.

This qualifier is normally used with the /FOREIGN and /BLOCKSIZE qualifiers to read or write fixed-length records on a block-structured device. In this case, the record size must be less than or equal to the block size that is specified or used by default. The block size may be in the range of 20 through 65,532 bytes if you are using RMS, or 18 through 65,534 bytes if you are not using RMS.

Use the /RECORDSIZE qualifier when mounting tapes without HDR2 labels (such as RT-11 tapes) to provide RMS with default values for the maximum record size and the length of the longest record.

/SHARE
/NOSHARE

Indicates, for a disk volume, whether the volume is shareable. If the volume has already been mounted for sharing by another user, and you request that it be mounted with the /SHARE qualifier, any other qualifiers you enter are ignored.

By default, the MOUNT command assumes that a volume is not shareable, and allocates the device on which it is mounted.

If you have previously allocated the device and specify the /SHARE qualifier, the MOUNT command deallocates the device so that other users can access it.

/SYSTEM

Makes the volume public, that is, available to all users in the system, as long as the UIC-based volume protection allows them access.

The logical name for the device is placed in the system logical name table. You must have the user privilege to place a name in the system logical name table (SYSNAM) to use the /SYSTEM qualifier.

/UNLOCK

Requests, for disk volumes, write access to the index file on the volume.

The /UNLOCK qualifier is valid only if the volume is mounted /NOSHARE.

/WINDOWS=n

Specifies the number of mapping pointers to be allocated for file windows. When a file is opened, the file system uses the mapping pointers to access data in the file.

You can specify a value from 7 through 80 to override the default value specified when the volume was initialized.

You must have the operator user privilege (OPER) to use the /WINDOWS qualifier.

/WRITE
/NOWRITE

Controls whether the volume can be written.

By default, a volume is considered read/write when it is mounted. You can specify /NOWRITE to provide read-only access to protect files. It is equivalent to write-locking the device.

Examples

1.
```
$ MOUNT   MT:  -
$_MATH06  STAT_TAPE
%MOUNT-I-MOUNTED, MATH06 mounted on _MTA0:
$ COPY    ST061178.DAT   STAT_TAPE:
```

The MOUNT command requests that the magnetic tape whose volume label is MATH06 be mounted on the device MTA0 and assigns the logical name STAT_TAPE to the volume.

Subsequently the COPY command copies the disk file ST061178.DAT to the tape.

2.
```
$ ALLOCATE DM:
  _DMB2:  ALLOCATED
$ MOUNT DMB2:  TEST_FILES
%MOUNT-I-MOUNTED, TEST_FILES mounted on _DMB2:
```

The ALLOCATE command requests an available RK06/RK07 device. After noting the device name in the response from the ALLOCATE command, the physical volume can be placed on the device. Then, the MOUNT command mounts the volume.

3.
```
$ MOUNT/SHARE DMA2:  PUB_FILES -
$_/PROTECTION=(SYSTEM:RWED,OWNER:RWED,GROUP:RWED)
```

The MOUNT command mounts the volume PUB_FILES as a shareable volume. The /PROTECTION qualifier allows all types of access to the system, the owner, and to other members of the volume owner's group. All other users are denied all types of access.

4.
```
$ MOUNT/FOREIGN/PROTECTION=(SYSTEM:RWPL,OWNER:RWPL,G:RW,W)  MTA1:
%MOUNT-I-MOUNTED, TESTER mounted on _MTA1:
$ MCR FLX
```

The MOUNT command requests the mounting of a foreign volume on the device MTA1. The system and owner categories receive read, write, physical and logical access. However, the group category receives only read and write access while the world category is denied access. The MCR FLX command invokes the RSX-11M File Transfer Utility to process the volume.

5.
```
$ MOUNT   DMA1:   PUBS_BACK
%SYSTEM-F-INCVOLLABEL, incorrect volume label
-MOUNT-I-VOLIDENT, label='BACK_UP_GMB', owner='MALCOLM',
 format= 'DECFILE11B'
```

The system response to the first MOUNT command indicates that the label was incorrectly specified. The second message from the MOUNT command displays the label on the volume, the volume owner, and the volume format (DECFILE11B indicates that the volume is in DIGITAL's Files-11, Structure Level 2 format).

6.
```
$ MOUNT/SYSTEM/BIND=MASTER_PAY -
$_DB1,DB2,DB3      PAYVOL1,PAYVOL2,PAYVOL3
```

The MOUNT command creates the volume set named MASTER_PAY consisting of the initialized volumes labelled PAYVOL1, PAYVOL2, and PAYVOL3. These volumes are currently mounted physically on the devices named DB1, DB2, and DB3, respectively. The volume PAYVOL1 is the root volume of the set.

The volumes are mounted as system volumes to make them available to all users.

7.
```
$ MOUNT/GROUP DB1:, DB2:, DB3:  -
$_PAYVOL1,PAYVOL2,PAYVOL3 PAY
```

The MOUNT command mounts and makes available on a group basis the volume set consisting of volumes labeled PAYVOL1, PAYVOL2, and PAYVOL3. The logical name PAY is assigned to the set; all users wishing to access files on these volumes can refer to the set as PAY:.

8.
```
$ MOUNT/GROUP/BIND=MASTER_PAY -
$_DB4:  PAYVOL4
```

The MOUNT command adds the volume labeled PAYVOL4 to the existing volume set MASTER_PAY. The root volume for the volume set must be online when this command is issued.

9.
```
$ MOUNT  DM1:  FRED -
$_/CACHE=(EXTENT=10,LIMIT=20,NOFILE_ID,WRITETHROUGH)
```

The volume labelled FRED is mounted on device DM1. Extent caching is enabled for 10 entries containing up to 2 percent of the current free space. File identification caching is disabled, as is writeback caching.

ON

Defines the default courses of action when a command or program executed within a command procedure (1) encounters an error condition or (2) is interrupted by CTRL/Y. The specified actions are taken only if the command interpreter is enabled for error checking or CTRL/Y interrupts; these are the default conditions.

Formats

```
ON   severity-level   THEN   [$] command
```

Command Qualifiers	Defaults
None.	ON ERROR THEN $ EXIT

```
ON CONTROL_Y   THEN   [$] command
```

Command Qualifiers	Defaults
None.	None.

Prompts

None.

Command Parameters

severity-level

Specifies the severity condition that will cause the action indicated to be taken. The severity-level is represented by one of the following keywords:

WARNING
ERROR
SEVERE_ERROR

You can truncate any of these keywords to one or more characters.

command

Specifies the action to be taken when errors equal to or greater than the specified level of error occur. You can specify any valid command line after the keyword THEN; you can optionally precede the command line with a dollar sign.

Description

For a description of the ON command and additional details on error and CTRL/Y handling, see the VAX/VMS Guide to Using Command Procedures.

Examples

1.

```
$ ON SEVERE_ERROR THEN CONTINUE
```

After this statement is executed in a command procedure, execution of the procedure continues when any warning, error, or severe error condition occurs. Once the statement has been executed as a result of a fatal error condition, the default action, 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.

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. When CTRL/Y is pressed, the GOTO command that transfers control to the line labeled CTRL_EXIT is executed. At this label, the procedure performs clean-up operations. In this example, clean-up consists of closing files and exiting.

OPEN

Opens a file for reading or writing at the command level.

Format

```
    OPEN   logical-name[:]  file-spec


    Command Qualifiers        Defaults

    /ERROR=label
    /READ                     /READ
    /WRITE                    /READ
```

Prompts

Log_Name: logical-name[:]

File: file-spec

Command Parameters

logical-name[:]

Specifies a logical name to be assigned to the file.

file-spec

Specifies the name of the file or device to be opened for input or output. If the file specification does not include a file type, the system uses the default file type of DAT.

No wild card characters are allowed in the file specification.

Description

A file can be opened for either reading or writing, but not both. After a file is opened, it is available for input or output at the command level with the READ and WRITE commands, or it can be accessed within a program image for input or output operations using the logical name.

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.

You can issue more than one OPEN command for the same file, and assign it different logical names. If you specify the same logical name on more than one OPEN command without closing the file, no warning message is issued and the file is not opened. If the file had been previously read, the next read request returns the record after the record previously read.

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.

For additional information on the VAX/VMS file handling commands, see the VAX/VMS Guide to Using Command Procedures.

Command Qualifiers

/ERROR=label

Specifies a label on a line in the command procedure to receive control if the open request results in an error. If no error routine is specified and an error occurs during the opening of the file, the command procedure continues execution at the next line in the file, as it does if no error occurs.

The error routine specified for this qualifier takes precedence over any action statement indicated in an ON command. If /ERROR is not specified, the current ON condition action is taken.

If an error occurs and a target label is successfully given control, the reserved global symbol $STATUS contains a successful completion status.

/READ

Requests that the file be opened for reading. This is the default, if neither /READ nor /WRITE is specified. You cannot open indexed sequential (ISAM) files for reading.

/WRITE

Requests that the file be opened for writing. If the file specification does not contain a version number and the file already exists, a new version of the file is created.

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 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
$ WRITE_LOOP:
    .
    .
    .

$ GOTO WRITE_LOOP
    .
    .
    .

$ OPEN_ERROR:
$ WRITE SYS$OUTPUT "Cannot open file TEMP.OUT"
$ EXIT
```

The OPEN command opens the file TEMP.OUT for output and assigns it the logical name OUTPUT_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.

PASCAL

Invokes the VAX-11 PASCAL[1] compiler to compile one or more source programs. This command is described in detail in the VAX-11 PASCAL User's Guide.

Format

```
PASCAL   file-spec[,...]
```

Command Qualifiers	Defaults
/[NO]CHECK	/NOCHECK
/[NO]CROSS_REFERENCE	/NOCROSS_REFERENCE
/[NO]DEBUG	/NODEBUG
/[NO]ERROR_LIMIT	/ERROR_LIMIT
/[NO]LIST[=file-spec]	(see text)
/[NO]MACHINE_CODE	/NOMACHINE_CODE
/[NO]OBJECT[=file-spec]	/OBJECT
/[NO]STANDARD	/STANDARD
/[NO]WARNINGS	/WARNINGS

Prompts

File: file-spec[,...]

Command Parameters

file-spec[,...]

Specifies one or more VAX-11 PASCAL source program files to be compiled. If you do not specify a file type for an input file, PASCAL uses the default file type of PAS.

You can specify more than one input file. If you separate the file specifications with commas (,), each file is compiled separately. If you separate the file specifications with plus signs (+), the files are concatenated and compiled as a single input file, producing single object and listing files. If you specify SYS$INPUT as the file-spec parameter, the source program must follow the command in the input stream. In this case, both the object module file (identified by the /OBJECT qualifier) and the listing file (identified by the /LIST qualifier) must be explicitly named.

No wild card characters are allowed in the file specification.

1. Available under separate license.

Command Qualifiers

/CHECK
/NOCHECK

Indicates whether the compiler should generate code to perform run-time checks. The code to be generated would check for illegal assignments to sets and subranges and out-of-range array bounds and case labels. If any of these conditions are detected at run-time, appropriate error messages are generated.

The default is /NOCHECK, which suppresses check code generation.

/CROSS_REFERENCE
/NOCROSS_REFERENCE

Controls whether the compiler creates a cross-reference listing as part of the listing file.

The default is /NOCROSS_REFERENCE, which does not create a cross-reference listing.

This qualifier is ignored if no listing file is being generated.

/DEBUG
/NODEBUG

Controls whether the compiler makes local symbol table and traceback information available to the VAX-11 Symbolic Debugger and the run-time error-reporting mechanism. If you specify /DEBUG, the compiler generates some DEBUG and TRACEBACK records for each procedure or program for which it is in effect.

The default is /NODEBUG, which produces only traceback information.

/ERROR_LIMIT
/NOERROR_LIMIT

Indicates whether the compiler should terminate after finding 30 errors (excluding warnings).

If you specify /NOERROR_LIMIT, the compilation continues until 500 errors are detected.

The default is /ERROR_LIMIT.

/LIST[=file-spec]
/NOLIST

Controls whether the compiler creates a listing file.

If you issue the PASCAL command from interactive mode, the compiler, by default, does not produce a listing file. However, if the PASCAL command is executed from batch mode, /LIST is the default. In either case, the listing file is not automatically printed. You must use the PRINT command to obtain a line printer copy of the PASCAL listing file.

Note that the /LIST qualifier permits you to include a file specification for the listing file. If you omit the file specification, the compiler defaults to the name of the first input source file, the default directory, and a file type of LIS.

No wild card characters are allowed in the file specification.

/MACHINE_CODE
/NOMACHINE_CODE

Specifies whether the listing file should include a representation of the machine code generated by the compiler. If no listing file is being generated, this qualifier is ignored.

The default is /NOMACHINE_CODE.

/OBJECT[=file-spec]
/NOOBJECT

Controls whether the compiler creates an object module. The /NOOBJECT qualifier is useful when you want to test the source program for compilation errors.

By default, the compiler produces an object module with the same file name as the first source file and a file type of OBJ.

No wild card characters are allowed in the file specification.

/STANDARD
/NOSTANDARD

Indicates whether the compiler prints messages if nonstandard PASCAL features are used in the source program. Nonstandard PASCAL features are the extensions to the PASCAL language that are incorporated in VAX-11 PASCAL.

The default is /STANDARD; if you specify /NOSTANDARD, no messages are printed.

/WARNINGS
/NOWARNINGS

Controls whether the compiler prints warning-level (W) diagnostic messages.

By default, the compiler generates warning messages; use the /NOWARNINGS qualifier to suppress these messages.

Note that messages generated when the /STANDARD qualifier is enabled appear even if /WARNINGS is disabled.

Examples

1.
```
$ PASCAL/LIST PROGA,PROGB,PROGC
```

   The source files PROGA.PAS, PROGB.PAS, and PROGC.PAS are compiled as separate files. The compiler produces three object files named, by default, PROGA.OBJ, PROGB.OBJ, and PROGC.OBJ. In addition, three listing files are generated that include the warning-level messages but exclude both cross-reference and machine code listings. The listing files are named PROGA.LIS, PROGB.LIS, and PROGC.LIS, respectively. Furthermore, the other defaults dictate that compilation stops after 30 errors are detected, warning messages are issued if nonstandard PASCAL statements are found, no run-time checks are made and only traceback records are generated.

PASSWORD

Specifies the password associated with the user name specified on a JOB card for a batch job submitted through the card reader.

Format

```
$ PASSWORD   password

Command Qualifiers                          Defaults

None.                                       None.
```

Prompts

None.

Command Parameters

password

Specifies the 1- through 31-character password associated with the user name specified on the JOB command.

Description

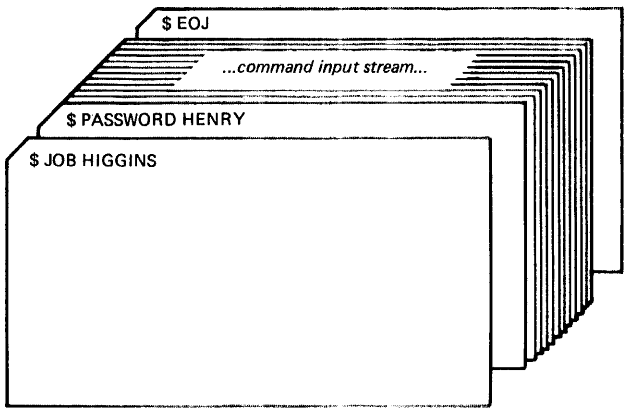
The PASSWORD command is valid only in a batch job submitted through the card reader. The $ is required. The password is checked against the password associated with the user name specified on the JOB card, and must be correct or the job is rejected.

Note that you may 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.

For more information on how to use commands like this one in command procedures, consult the VAX/VMS Guide to Using Command Procedures.

Examples

1.



The JOB and PASSWORD commands precede a batch job submitted from the card reader. An EOJ command marks the end of the job.

Invokes the VAX-11 Image File Patch Utility (PATCH) to patch an executable image, shareable image, or device driver image. This utility is described in detail in the VAX-11 PATCH Utility Reference Manual.

Format

```
    PATCH    file-spec

    Command Qualifiers                      Defaults

    /JOURNAL[=file-spec]                    None.
    /OUTPUT[=file-spec]
    /UPDATE=[(eco-level[,...])]
    /VOLUME[=n]
```

Prompts

File: file-spec

Command Parameters

file-spec

Specifies the image file to be patched or a command procedure that contains both the name of the image file to be patched and PATCH commands.

If the file specification denotes an image file, the file specification must contain the file name. If you omit the remaining fields (device, directory, file type, and version number), PATCH uses your default device and directory, assumes a file type of EXE, and uses the highest version of the image file.

If the file specification denotes a command procedure, the file-spec parameter must be preceded by an at sign (@). Only the file name is required. If you omit the remaining fields (device, directory, file type, and version number), PATCH uses your default device and directory, assumes a file type of COM, and locates the highest version of the command procedure.

No wild card characters are allowed in the file specification.

Description

The VAX-11 Image File Patch Utility (PATCH) is an editing tool used by system programmers to facilitate the maintenance of executable images, shareable images, and device driver images. After a program has been compiled or assembled and linked, it may need to be modified. Using PATCH, you can make the necessary modifications, in the form of patches, without the inconvenience of a lengthy editing session, nor the need to recompile or reassemble and relink.

You can use PATCH to edit files interactively or through command procedures. When you edit interactively, input consists of an input image file and PATCH commands. When you edit using command procedures, input consists of an input image file and a command procedure that contains the PATCH commands you want processed.

PATCH creates one or more of the following files as output:

Output image file	The updated input image file. This file must be explicitly requested using the UPDATE command provided within PATCH.
Journal file	An ASCII file that contains a record of the PATCH session. This file is automatically created.
Command procedure	A file that contains all successful PATCH commands. This file must be explicitly requested using the CREATE command provided within PATCH.

For a detailed description of the PATCH utility commands, see the VAX-11 PATCH Utility Reference Manual.

Command Qualifiers

/JOURNAL[=file-spec]

Indicates the journal file specification. If you omit fields in the file specification, PATCH supplies the following default values:

Field	**Default Value**
device and directory	current process's defaults
file name	name of input image file
file type	JNL
version	1

Subsequent PATCH sessions append information to the journal file, rather than create a new version of this file.

No wild card characters are allowed in the file specification.

/OUTPUT[=file-spec]

Indicates the output image file specification.

If you omit fields in the file specification, PATCH supplies the following default values:

Field	**Default Value**
device and directory	current process's defaults
file name	name of input image file
file type	EXE
version	one greater than the most recent copy of the input image file

The output image file is not created unless you issue the PATCH UPDATE command at the end of the PATCH session. You can issue multiple PATCH UPDATE commands in a single session. The first UPDATE command creates the output image file; subsequent UPDATE commands overwrite this file. See the VAX-11 PATCH Utility Reference Manual.

No wild card characters are allowed in the file specification.

/UPDATE[=(eco-level[,...])]

Requests that only the patches associated with the specified ECO levels be processed. If you specify more than one ECO level, you must separate the ECO levels by commas and enclose the list in parentheses.

When you specify the /UPDATE qualifier, the PATCH command file specification denotes either a command procedure that contains the patches to be processed or an image file to which certain patches are applied. When the file specification denotes a command procedure, the /UPDATE qualifier must precede the file specification on the command line. When the file specification denotes an image file, the /UPDDATE qualifier can precede or follow the file specification. In either case, the file specification is required.

If PATCH encounters an ECO level in a command procedure that does not match the ECO level specified on the /UPDATE qualifier, PATCH ignores the ensuing patch but displays a message. Whenever you omit the optional ECO levels, PATCH responds by processing all patches submitted.

/VOLUME[=n]

Requests that the output file be placed on a specified relative volume number of a multivolume set.

If you specify /VOLUME without a number, the number defaults to the relative volume number of the input image file.

If the /VOLUME qualifier is not specified, the file is placed in an arbitrary position within the multivolume set.

Examples

1.
```
$ PATCH AVERAGE /JOURNAL=TEST /OUTPUT=TEST
```

   This command invokes PATCH for an interactive PATCH session with the image file AVERAGE.EXE. The journal file and output image file created by this session are both named TEST and reside in the default device and directory.

2.
```
$ PATCH /UPDATE=(100,102) @ORION
$
```

   This PATCH command executes the command procedure PATCHCMD.COM in interactive mode. The /UPDATE qualifier requests that only the patches identified by the ECO levels 100 and 102, contained in ORION.COM, be processed. The first record in ORION.COM must specify the input image file. The second DCL prompt ($) indicates that the patches were successfully applied.

3.
```
$ PATCH /UPDATE=(101,103) DIAG


PATCH BASE LEVEL  V2.22 28-JUN-1979

PATCH> SET ECO 101
PATCH> EXAMINE 400
00000400:     7681AD01
PATCH> DELETE 400 = 7681AD01
old:      00000400:      7681AD01
new:      00000400:      00000000
PATCH> UPDATE
UPDATING IMAGE FILE "DB1:[SYSEXE]DIAG.EXE;7"
PATCH> SET ECO 102
%PATCH-I-UPDATE, patch with eco level 102 ignored due to update qualifier
PATCH> SET ECO 103
PATCH> EXAMINE/INSTRUCTION 627
00000627:     PUSHAL L^00000224
PATCH> INSERT/INSTRUCTION 627 = "PUSHAL L^00000224"
NEW> "MOVL (R6),(R5)"
NEW> EXIT
old:      00000627:      PUSHAL  L^00000224
new:      00000627:      BRW     00007806
new:      0000062A:      NOP
new:      0000062B:      NOP
new:      0000062C:      NOP
new:      00007806:      PUSHAL  L^00000224
new:      00007809:      BRW     0000062D
PATCH> UPDATE
UPDATING IMAGE FILE "DB1:[SYSEXE]DIAG.EXE;7"
PATCH> EXIT
$
```

In this example, PATCH is invoked for an interactive PATCH session with the image file DIAG.EXE. The /UPDATE qualifier requests that only the patches associated with the ECO levels 101 and 103 be applied to DIAG.EXE. During the course of the PATCH session, the ECO level 102 is attempted to be defined, but because this level was not specified with the /UPDATE qualifier, PATCH displays a message indicating that the patch pertaining to ECO level 102 was ignored. All commands issued between the SET ECO 102 command and the SET ECO 103 command would be ignored.

PRINT

Queues one or more files for printing, either on a default system printer or on a specified device.

Format

```
PRINT   file-spec[,...]

Command Qualifiers                Defaults

/AFTER=absolute-time
/CHARACTERISTICS=(c[,...])
/DEVICE=device-name[:]
/FORMS=type
/[NO]HOLD                         /NOHOLD
/[NO]IDENTIFY                     /IDENTIFY
/JOB_COUNT=n                      /JOB_COUNT=1
/[NO]LOWERCASE                    /NOLOWERCASE
/NAME=job-name
/PRIORITY=n
/QUEUE=queue-name[:]

File Qualifiers                   Defaults

/[NO]BURST                        /NOBURST
/COPIES=n                         /COPIES=1
/[NO]DELETE                       /NODELETE
/[NO]FEED                         /FEED
/[NO]FLAG_PAGE
/[NO]HEADER                       /NOHEADER
/PAGE_COUNT=n
/SPACE[=n]
```

Prompts

```
File:  file-spec[,...]
```

Command Parameters

file-spec[,...]

Specifies one or more files to be printed. If you specify more than one file, separate the file specifications with either commas (,) or plus signs (+). In either case, the PRINT command concatenates the files into a single print job.

You can use wild card characters in the directory specification, file name, file type, or version number fields. See Section 2.1.6.

If you do not specify a file type for the first input file, the PRINT command uses the default file type of LIS.

Note that the PRINT command cannot print a file that resides on a device that is allocated.

Description

A file or files queued by the PRINT command are considered a job. The system assigns a unique job identification number to each job in the system and displays this job number when the PRINT command completes execution.

Printer queues are identified by name; if you do not specify a queue name with the /DEVICE or /QUEUE qualifiers, the system queues the job to an available queue. If no physical printer is currently available with the required characteristics, or if the job is in a hold status, the job is placed in the queue named SYS$PRINT. The PRINT command, by default, displays the name of the queue on which it entered the job.

Command Qualifiers

/AFTER=absolute-time

Requests that the file not be printed until a specific time of day.

Specify the time using the standard syntax rules for specifying absolute time values given in Section 5.8.

If the specified time has already passed, the file is queued for printing immediately.

/CHARACTERISTICS=(c[,...])

Specifies one or more characteristics desired for printing the file(s). If you specify more than one characteristic, separate them by commas and enclose the list in parentheses. Codes for characteristics are installation defined. For details on specifying characteristics, see the VAX/VMS System Manager's Guide.

/DEVICE=device-name[:]

Requests that the file(s) specified be queued for printing on a specific device to which queueing is allowed. If the /DEVICE qualifier is not specified, files are queued, by default, to SYS$PRINT. This qualifier is synonymous with the /QUEUE qualifier.

/FORMS=type

Specifies the forms type required for the specified file(s).

Specify the forms type using a numeric value or alphanumeric code. Codes for forms types are installation-defined. Forms type control is discussed in the VAX/VMS System Manager's Guide.

/HOLD
/NOHOLD

Controls whether the file is available for printing immediately. When you specify the /HOLD qualifier, the file is not released for actual printing until you use the SET QUEUE/ENTRY command to release it.

By default the job is not held before printing.

/IDENTIFY
/NOIDENTIFY

Controls whether the PRINT command displays a message indicating the job number of the print job and the name of the queue in which it is entered.

By default, the PRINT command displays this information whenever a job is successfully queued.

/JOB_COUNT=n

Requests that the entire job be printed n times, where n is a decimal number from 1 through 255.

By default the job is printed one time.

/LOWERCASE
/NOLOWERCASE

Indicates whether the specified file(s) contain lowercase alphabetic letters and must be printed on an available printer that can print uppercase and lowercase letters.

The default is /NOLOWERCASE, which means files may be printed on printers supporting only uppercase letters.

/NAME=job-name

Defines a 1- through 8-alphanumeric character name string to identify the job. The job-name is displayed by the SHOW QUEUE command and is printed in the top and bottom rows of the flag page for the job.

If you do not specify /NAME, the name string defaults to the file name (truncated to eight characters, if necessary) of the first, or only, file in the job.

/PRIORITY=n

Specifies the priority of the print job. The priority, n, must be in the range of 0 through 31, where 0 is the lowest priority and 31 is the highest.

By default, jobs are assigned the same priority as your current process priority. The user privilege ALTPRI is required to set a priority value that is higher than your current process's priority.

/QUEUE=queue-name[:]

Requests that the specified file(s) be printed on a specific device. This qualifier is synonymous with the /DEVICE qualifier.

File Qualifiers

/BURST
/NOBURST

Controls whether a burst page is included on output. A burst page precedes a flag page and contains the same information. However, it is printed over the perforation between pages. Note that when you specify /BURST you want FLAG_PAGE enabled so that a flag page will automatically follow the burst page. (Flag pages are generally enabled when a device queue is initialized. If not, you can enable flag pages with the /FLAG_PAGE qualifier for the PRINT command.)

Use the /BURST qualifier to override the installation-defined defaults set up for printers when they are started.

/COPIES=n

Specifies the number of copies to print. By default, the PRINT command prints a single copy of a file; you can use /COPIES to request up to 255 copies.

If you specify /COPIES 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.

/DELETE
/NODELETE

Controls whether files are deleted after printing. If you specify /DELETE after the PRINT command name, all files specified 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.

By default, files are not deleted after printing.

/FEED
/NOFEED

Controls whether the PRINT command automatically inserts form feeds when it nears the end of a page. By default, the PRINT command inserts a form feed when the printer is within four lines of the end of the form. For example, on standard 66-line forms, a form feed occurs after 62 lines are printed. 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.

Files that do not have RMS record attributes of FORTRAN carriage control (FTN), implied carriage return (CR), or print file format (PRN) are considered to be internally formatted files. Neither the /FEED nor /NOFEED qualifiers have any affect on internally formatted files.

/FLAG_PAGE
/NOFLAG_PAGE

Controls whether a flag page is printed preceding output. Use this qualifier to override the installation-defined defaults set up for printers when they are started.

If you specify the /FLAG_PAGE qualifier as a command qualifier with the command name, a flag page is printed for the entire job.

If you specify the /FLAG_PAGE qualifier with any file specification, a separate flag page is printed preceding the associated file.

/HEADER
/NOHEADER

Controls whether the name of the file is printed at the top of each output page. By default, the file specification is printed only at the top of the first page of output.

/PAGE_COUNT=n

Specifies the number of pages of the specified file to print. You can only use /PAGE_COUNT to qualify file specifications; it cannot qualify the command name. You can use the /PAGE_COUNT qualifier to print just a few pages of a large file.

By default, all pages are printed.

/SPACE[=n]

Specifies the number of spaces (that is, blank lines) to leave between lines of output on the specified file(s). You can specify 1 or 2 to indicate the number of blank lines.

If you do not specify /SPACE, no extra lines are printed between lines of output; if you specify /SPACE without a value for n, the output is double-spaced.

Examples

1.
```
$ PRINT AVERAGE
  Job 236 entered on queue LPA1:
```

The PRINT command queues the file AVERAGE.LIS for printing on a system printer. The system displays the job number and the name of the printer to which it queued the file.

2.
```
$ PRINT  ALPHA.TXT+BETA+GAMMA
  Job 237 entered on queue LPA1:
```

The PRINT command prints the files ALPHA.TXT, BETA.TXT, and GAMMA.TXT as a single file. The printer output file is named ALPHA.TXT.

3.
```
$ ASSIGN  LPA0:  SYS$PRINT
$ PRINT *.TXT
  Job 238 entered on queue LPA0:
```

The ASSIGN command creates an entry in the process logical name table for the logical name SYS$PRINT to override the system logical name definition. Subsequently, a PRINT command translates the logical name SYS$PRINT and queues the job on the queue LPA0. This job consists of the highest versions of all files with the file type of TXT.

4.
```
$ PRINT/COPIES=10/AFTER=20   ALPHA.TXT
  Job 237 entered on queue LPA1:
```

The PRINT command queues 10 copies of the file ALPHA.TXT to the printer, but requests that the copies not be printed until after 8:00 P.M.

5.
```
$ PRINT/LOWERCASE   ALPHA.TXT/COPIES=2, -
$_BETA.DOC/COPIES=3
  Job 238 entered on queue LPA1:
```

The print job queued by this print command 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.

6.
```
$ PRINT/JOB_COUNT=3   ALPHA.TXT,BETA/NOIDENTIFY
```

This PRINT command 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 number not be displayed.

7.
```
$ PRINT/HOLD   MASTER.DOC
  Job 240 entered on queue SYS$PRINT
    .
    .
    .

$ SET QUEUE/ENTRY=240/RELEASE
```

The PRINT command 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.

Deletes all but the highest-numbered version or versions of a specified file or files.

Format

```
PURGE    [file-spec[,...]]

Command Qualifiers         Defaults

/KEEP=n                    /KEEP=1
/[NO]LOG                   /NOLOG
```

Prompts

None.

Command Parameters

file-spec[,...]

Specifies one or more files to be purged. If you specify more than one file, separate them with either commas (,) or plus signs (+). If you do not provide a file specification, the PURGE command purges all files in the current default directory.

The PURGE command does not provide file name or file type defaults; version numbers are not allowed. You can use wild card characters in the directory specification, file name or file type fields. See Section 2.1.6.

Command Qualifiers

/KEEP=n

Specifies the maximum number of versions to retain of the specified file(s).

If you do not specify /KEEP, all but the highest-numbered version of the specified file(s) are deleted.

/LOG
/NOLOG

Controls whether the PURGE command displays the file specifications of files as it deletes them.

By default, PURGE does not display the file specifications of files it purges.

Examples

1.
```
$ PURGE *.COM
```

The PURGE command deletes all but the highest-numbered version of each file with a file type of COM.

2.
```
$ PURGE AVERAGE.FOR/KEEP=2
```

The PURGE command deletes all but the two highest numbered versions of the file AVERAGE.FOR.

3.
```
$ PURGE
```

The PURGE command deletes all but the highest-numbered version of all files in the default directory.

4.
```
$ PURGE [MAL.TESTFILES]/LOG
%PURGE-I-FILPURGED, DBA1:[MAL.TESTFILES]AVE.OBJ;1 deleted
%PURGE-I-FILPURGED, DBA1:[MAL.TESTFILES]BACK.OBJ;2 deleted
```

The PURGE command purges all files cataloged in the subdirectory named MAL.TESTFILES. The /LOG qualifier requests the PURGE command to display the specification of files it deletes.

Reads a single record from a specified input file and assigns the contents of the record to a specified symbol name.

Format

```
READ   logical-name[:]  symbol-name

Command Qualifiers                    Defaults

/END_OF_FILE=label                    None.
/ERROR=label
```

Prompts

Log_Name: logical-name[:]

Symbol: symbol-name

Command Parameters

logical-name[:]

Specifies the logical name of the input file from which a record is to be read. The OPEN command assigns a logical name to a file and places the name in the process logical name table.

If the READ command is executed interactively and the logical name is specified as SYS$INPUT, SYS$OUTPUT, SYS$COMMAND, or SYS$ERROR, the command interpreter prompts for input data.

symbol-name

Specifies a 1- through 255-alphanumeric character symbol name to be equated to the contents of the record being read. The symbol is always defined locally to the current command level. If the symbol is already defined, the READ command redefines it to the new value.

Description

The READ command can read data only from sequential files or devices. After each record is read from the specified file, the READ command positions the record pointer at the next record in the file.

The maximum size of any record that can be read in a single READ command is 255 bytes.

Before a file can be read, it must be opened with the OPEN command and assigned a logical name. 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.

For additional information on the VAX/VMS file handling commands, see the VAX/VMS Guide to Using Command Procedures.

Command Qualifiers

/END_OF_FILE=label

Specifies the label on a line in the current command procedure. When the last record in the file is read, the command interpreter transfers control to the command line at the specified label.

If /END_OF_FILE is not specified, then control is given to the error label specified with the /ERROR qualifier, if any, when the end-of-file is reached. If neither /ERROR nor /END_OF_FILE is specified, then the READ command that results in an end-of-file condition returns with an error status code.

/ERROR=label

Specifies a label on a line in the command procedure to receive control if the read request results in an error. If no error routine is specified and an error occurs during the reading of the file, the command procedure continues execution at the next line in the file, as it does if no error occurs.

The error routine specified for this qualifier takes precedence over any action statement indicated in an ON command. 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 reserved global symbol $STATUS contains a successful completion value.

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 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:
   .
   .
   .

  error handling
$ OKAY:
$ CLOSE MSGFILE
$ EXIT
```

The READ command specifies labels to receive control at the end-of-file and on error conditions. At the end-of-file, the procedure uses the CLOSE command to close the file and exits. When an error occurs, the procedure closes the file.

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.

RENAME

Changes the directory specification, file name, file type, or file version of an existing disk file or disk directory.

Format

```
RENAME   input-file-spec[,...]  output-file-spec

Command Qualifiers        Defaults

/[NO]CONFIRM              /NOCONFIRM
/[NO]LOG                  /NOLOG
/[NO]NEW_VERSION          /NEW_VERSION
```

Prompts

From: input-file-spec[,...]

To: output-file-spec

Command Parameters

input-file-spec[,...]

Specifies the names of one or more files whose specifications are to be changed.

You can use wild card characters in the directory specification, file name, file type, or version number fields of the file specification. See Section 2.1.6. In this case, all files whose specifications satisfy the fields that are specified 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 specification to provide defaults for nonspecified fields in the output file.

You can specify an asterisk (*) in place of the directory specification, file name, file type, or version number of the output-file-spec parameter; the RENAME command uses the corresponding field in the input file specification to name the output file. Wild card characters in corresponding fields of the input and output file specification result in multiple rename operations.

The RENAME command supplies output file version numbers according to the first description below that applies:

1. If the output file specification contains an explicit version number, the RENAME command uses that version number.

2. If the input file specification or output file specification contains an asterisk in the version number field, the RENAME command uses the version number of each input file to name the output file.

3. If no file currently exists with the same file name and file type as that specified for the output file, the RENAME command gives the file a version number of 1.

4. If a file currently exists with the same file name and file type as that specified for the output file, the RENAME command gives the output file a version number 1 greater than the highest existing version, unless /NONEW_VERSION is specified.

Command Qualifiers

/CONFIRM
/NOCONFIRM

Controls whether the RENAME command displays the file specification of each file before renaming and requests you to confirm whether or not the file actually should be renamed. If you specify /CONFIRM, you must respond to a prompt with a Y (YES) or a T (TRUE), followed by a carriage return, before the RENAME command renames the file. If you enter anything else, such as N or NO, the file is not renamed.

By default, the RENAME command does not request confirmation of files it is renaming.

/LOG
/NOLOG

Controls whether the RENAME command displays the file specification of each file that it renames.

By default, the RENAME command does not display the names of files after it renames them.

/NEW_VERSION
/NONEW_VERSION

Controls whether the RENAME command automatically assigns a new version number to the output file, if a file with the same file name and file type already exists.

The default is /NEW_VERSION; if a file with the same file name and file type exists, the RENAME command assigns a new version number.

Examples

1. `$ RENAME  AVERAGE.OBJ  OLDAVE`

   The RENAME command changes the file name of the highest existing version of the file AVERAGE.OBJ to OLDAVE.OBJ. If no file named OLDAVE.OBJ currently exists, the new file is given a version number of 1.

2. `$ RENAME/NONEW_VERSION  SCANLINE.OBJ;2   BACKUP.OBJ`

   The RENAME command 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 reports the error.

3. `$ RENAME  *.TXT;*   *.OLD;*`

   The RENAME command renames all versions of all files with file types of TXT to have file types of OLD. The file names and version numbers are not changed.

4. `$ RENAME  [HIGGINS]COMPLIB.OLB  [HIGGINS.LIBRARY]`

   The RENAME command changes the directory name of the object module library COMPLIB.OLB. The library is now cataloged in the subdirectory [HIGGINS.LIBRARY].

5. `$ RENAME  [MALCOLM.TESTFILES]SAVE.DAT   []TEST`

   The RENAME command renames the file SAVE.DAT in the directory MALCOLM.TESTFILES to TEST.DAT. The new file is cataloged in the current default directory.

6.
```
$ RENAME/LOG
$_From:      DATA.*,INFO.*
$_To:        NEW
%RENAME-I-RENAMED, _DMA0:[SYSTE]DATA.AAA;1 renamed to _DMA0:[SYSTE]NEW.AAA;1
%RENAME-I-RENAMED, _DMA0:[SYSTE]DATA.BBB;1 renamed to _DMA0:[SYSTE]NEW.BBB;1
%RENAME-I-RENAMED, _DMA0:[SYSTE]DATA.CCC;1 renamed to _DMA0:[SYSTE]NEW.CCC;1
%RENAME-I-RENAMED, _DMA0:[SYSTE]INFO.001;1 renamed to _DMA0:[SYSTE]NEW.001;1
%RENAME-I-RENAMED, _DMA0:[SYSTE]INFO.002;1 renamed to _DMA0:[SYSTE]NEW.002;1
%RENAME-I-RENAMED, _DMA0:[SYSTE]INFO.003;1 renamed to _DMA0:[SYSTE]NEW.003;1
$
```

   Three files exist with the file name of DATA and three files have the file name of INFO. This RENAME command illustrates wild card 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.

REQUEST

Displays a message at a system operator's terminal, and optionally requests a reply. System operators are identified by the function(s) they perform; if more than one operator is designated for a particular function, all receive the specified message.

Format

```
REQUEST   "message-text"

Command Qualifiers            Defaults

/REPLY
/TO[=(operators[,...])]       (see text)
```

Prompts

None.

Command Parameters

"message-text"

Specifies the text of a message to be displayed at the specified operator's terminal(s).

The message text can have a maximum of 128 characters; if you type more than one word, enclose the text in quotation marks (").

Description

When you use the REQUEST command to send a message to an operator, the message is displayed at one or more operators' terminals, according to the keywords specified with the /TO qualifier. If you specify /REPLY, the message is assigned an identification number, so that the operator can respond to the message.

If you specify /REPLY; you receive the message:

```
%OPCOM-S-OPRNOTIF, operator notified, waiting...hh:mm:ss
```

When the operator responds to your request, you receive a message in the format:

```
%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, you receive the message:

```
REQUEST - Enter message or cancel with <^Z>
REQUEST - Message?
```

At this time, you can enter either a message to be displayed at the specified operator(s)' terminal(s), or you can press CTRL/Z to cancel the request. If you enter a 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 initialized.

Command Qualifiers

/REPLY

Requests a reply to the specified message.

If you request a reply, the message is assigned a unique identification so that the operator can respond.

/TO[=(operators[,...])]

Specifies one or more operators to whom you wish to send the message. You can specify one or more of the following keywords. If you specify more than one keyword, separate them with commas and enclose the list in parentheses. By default, the message is sent to all terminals currently designated as operators' terminals.

CARDS	Send the message to the card reader operator.
CENTRAL	Send the message to the central system operator.
DEVICES	Send the message to operators designated to mount disk and tape volumes.
DISKS	Send the message to operators designated to mount and dismount disk volumes.
NETWORK	Send the message to the network operator.
OPER1 through OPER12	Send the message to an installation-specified operator identified as OPR1, OPR2, and so on.
PRINTERS	Send the message to operators designated to respond to printer requests.
TAPES	Send the message to operators designated to mount and dismount tape volumes.

Examples

1.
```
$ PRINT/COPIES=2  REPORT.OUT/FORMS=H
  Job 401 entered on queue LPA1:
$ REQUEST/REPLY/TO=PRINTERS -
$_"Have queued job 401 as forms=H;  can you print it?"
%OPCOM-S-OPRNOTIF, operator notified, waiting...10:42:16.10
%OPCOM-S-RQSTCMPLTE, request complete
%OPCOM-S-OPREPLY, AFTER 11:00
```

The PRINT command requests that multiple copies of a file be printed using a special type of paper (/FORMS=H). After queueing the PRINT job, the REQUEST command sends a message to the system operator designated to handle users' requests about print jobs. The message asks the operator how long before the file will be printed.

The reply shows the text entered by the operator.

2.
```
$ REQUEST/REPLY  "ARE YOU THERE"
%OPCOM-S-OPRNOTIF, operator notified, waiting...14:54:30.33
^C
REQUEST-Enter message or cancel request with <^Z>
REQUEST-Message?^Z
%OPCOM-S-OPRNOTIF, operator notified, waiting... 14:59:01.38
%OPCOM-F-RQSTCAN, request was canceled
```

The REQUEST command issues a message to see if there are any operators. When no response is received, CTRL/C interrupts the request and then CTRL/Z cancels it.

RUN (Image)

Places an image into execution in the process.

You can truncate the RUN command to a single letter, R.

Format

```
RUN    file-spec

Command Qualifiers                      Defaults

/[NO]DEBUG                              None.
```

Prompts

File: file-spec

Command Parameters

file-spec

Specifies an executable image to be executed. If you do not specify a file type, the RUN command uses the default file type of EXE.

No wild card characters are allowed in the file specification.

Command Qualifiers

/DEBUG
/NODEBUG

Controls, for native VAX-11 images, whether the image is to be run with the debugger. If the image was linked with the /DEBUG qualifier and you do not want the debugger to prompt, use the /NODEBUG qualifier. 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, /DEBUG is invalid.

Examples

1. `$ RUN LIBRA`

   The image LIBRA.EXE starts executing in the process.

2.
   ```
   $ MACRO/ENABLE=DEBUG ORION
   $ LINK/DEBUG ORION
   $ RUN ORION

        VAX-11 DEBUG V2.0

   %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 prompt. If an error occurs while the image executes, the debugger can perform traceback and report on the error.

RUN (Process)

Creates a subprocess or a detached process to execute a specified image. If you specify any of the command qualifiers listed, the RUN command creates a process.

Format

```
RUN   file-spec


Command Qualifiers                   Defaults

/[NO]ACCOUNTING                      /ACCOUNTING
/AST_LIMIT=quota
/[NO]AUTHORIZE                       /AUTHORIZE
/BUFFER_LIMIT=quota
/DELAY=delta-time
/ERROR=file-spec
/FILE_LIMIT=quota
/INPUT=file-spec
/INTERVAL=delta-time
/IO_BUFFERED=quota
/IO_DIRECT=quota
/MAILBOX=unit
/MAXIMUM_WORKING_SET=quota
/OUTPUT=file-spec
/PAGE_FILE=quota
/PRIORITY=n
/PRIVILEGES=(privilege[,...])        /PRIVILEGES=SAME
/PROCESS_NAME=process-name
/QUEUE_LIMIT=quota
/[NO]RESOURCE_WAIT                   /RESOURCE_WAIT
/SCHEDULE=absolute-time
/[NO]SERVICE_FAILURE                 /NOSERVICE_FAILURE
/SUBPROCESS_LIMIT=quota
/[NO]SWAPPING                        /SWAPPING
/TIME_LIMIT=limit
/UIC=uic
/WORKING_SET=default
```

Prompts

File: file-spec

Command Parameters

file-spec

Specifies an executable image to be executed in a separate process. If the file specification does not include a file type, the RUN command uses the default file type of EXE.

No wild card characters are allowed in the file specification.

Description

When you use any of the qualifiers listed above with the RUN command, the RUN command creates a process and displays the process identification (PID) of the process on SYS$OUTPUT. The process executes the image specified in the file-spec parameter. When the image completes execution, the system deletes the process.

By default, the RUN command creates a subprocess with the same UIC, current disk and directory defaults, privileges, and priority of the current process.

The /UIC qualifier requests the RUN command to create a detached process; you must have the user privilege DETACH to create a detached process. When you create a detached process, the resource quotas are neither pooled nor deductible.

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 with 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
/PRIORITY
/PRIVILEGES
/PROCESS_NAME
/SERVICE_FAILURE
/SWAPPING
```

Assigning Resource Quotas: When you issue 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 quota below the minimum, the RUN command cannot create the process. To determine your current quotas, issue 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
/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 subprocess. 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
/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 a state of hibernation. The process cannot execute the image until it is awakened. Time values specified with these three qualifiers control when the process will be awakened to execute the specified image.

You can schedule wakeups for a specified delta time (/DELAY qualifier) or absolute time (/SCHEDULE qualifier). Optionally, you can 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. When the image completes normally, the process is returned to a state of hibernation. When the next scheduled wakeup occurs, the image is reactivated. Note that if the image completes abnormally, for example, if it calls the Exit system service, it does not return to hibernation.

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.

Command Qualifiers

/ACCOUNTING
/NOACCOUNTING

Controls whether accounting records are to be logged for the created process. By default, all processes are logged in the system accounting file.

You must have the user privilege ACNT to disable accounting.

/AST_LIMIT=quota

Specifies the maximum number of Asynchronous System Traps (ASTs) the created process can have outstanding.

If you do not specify an AST limit quota, the default value established at system generation time is used; the minimum required for any process to execute is 2. A value of 10 is typical.

This quota is nondeductible.

/AUTHORIZE
/NOAUTHORIZE

Controls, when the image to be executed is the system login image (LOGINOUT.EXE), whether login searches the user authorization file to validate a detached process.

By default, the login image checks the user authorization file whenever a detached process is created. Specify /NOAUTHORIZE to create a detached process running under the control of the command interpreter. The process permanent files specified by the /INPUT and /OUTPUT qualifiers are made available to the command interpreter for input and output.

The user privilege DETACH is required to create a detached process. Any nonspecified attributes of the created process default to the same as those of the current process.

/BUFFER_LIMIT=quota

Specifies the maximum amount of memory, in bytes, that the process may use for buffered I/O operations or 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 10240 is typical.

This quota is pooled.

/DELAY=delta-time

Requests that the created process be placed in hibernation, and awakened after a specified time interval has elapsed.

Specify the delta-time according to the rules for delta time specifications (these rules are given in Section 5.8.)

If you specify /INTERVAL with /DELAY, the first wakeup request occurs at the time specified by /DELAY and all subsequent wakeups occur at intervals as specified by /INTERVAL.

/ERROR=file-spec

Defines a 1- through 63-alphanumeric character equivalence name string 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.

/FILE_LIMIT=quota

Specifies the maximum number of files that a process can have open at any one time.

If you do not specify an open file quota for a created process, the system uses the default value established at system generation time. The minimum amount required for any process to execute is 2. A value of 20 is typical.

This quota is pooled.

/INPUT=file-spec

Defines a 1- through 63-alphanumeric character equivalence name string for the logical device name 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 5.8.

If you specify /DELAY or /SCHEDULE with /INTERVAL, the first wakeup occurs at the time specified by /DELAY or /SCHEDULE, and 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.

If the image to be executed is an RSX-11M image, the /INTERVAL qualifier has the effect of the /DELAY qualifier. Only one wakeup occurs.

/IO_BUFFERED=quota

Specifies the maximum number of system-buffered I/O operations the created process can have outstanding at a time.

If you do not specify a buffered I/O quota, the default value established at system generation time is used; the minimum required for any process to execute is 2. A value of 6 is typical.

This quota is nondeductible.

/IO_DIRECT=quota

Specifies the maximum number of direct I/O operations the created process can have outstanding at a time.

If you do not specify a direct I/O quota, the default value established at system generation time is used; the minimum required for any process to execute is 2. A value of 6 is typical.

This quota is nondeductible.

/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 process is deleted.

Mailbox creation and use and process termination mailboxes are described in the VAX/VMS System Services Reference Manual.

/MAXIMUM_WORKING_SET=quota

Specifies the maximum size to which the image to be 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).

If you do not specify a working set quota, the system uses the default value extablished at system generation time. The minimum value required for any process to execute is 10 pages. A value of 200 is typical.

This quota is nondeductible.

/OUTPUT=file-spec

Defines a 1- through 63-alphanumeric character equivalence name string for the logical device name SYS$OUTPUT. The logical name and equivalence 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; that is, the amount of secondary storage to use during the execution of the image.

If you do not specify a paging file quota, the system uses the default value established at system generation time. The minimum value required for a process to execute is 256 pages. A value of 10000 pages is typical.

This quota is pooled.

/PRIORITY=n

Specifies the base priority at which the created process is to execute.

The priority, n, is a decimal number from 0 through 31, where 31 is the highest priority and 0 is the lowest. Normal priorities are in the range 0 through 15, and real time priorities are in the range of 16 through 31.

You must have the ALTPRI user privilege to set the base priority higher than the priority of your current process.

If you specify a higher value when you do not have the privilege, or if you do not specify a priority, the priority defaults to the base priority of the current process.

/PRIVILEGES=(privilege[,...])

Defines user privileges for the created process. The privilege list consists of one or more of the keywords listed below. You may extend any privilege you possess to a process you create. However, you must have the SETPRV user privilege to give a process you create any privileges that you yourself do not have.

If you specify more than one privilege, separate the privileges by commas and enclose the list in parentheses.

Privilege	**Meaning**
[NO]ACNT	Allow/disallow the process to create processes for which no accounting messages are written
[NO]ALLSPOOL	Allow/disallow the process to allocate spooled devices
[NO]ALTPRI	Allow/disallow the process to set priority values
[NO]BUGCHK	Allow/disallow the process to make bug check error log entries
[NO]BYPASS	Allow/disallow the process to bypass UIC protection
[NO]CMEXEC	Allow/disallow the process to change its mode to executive
[NO]CMKRNL	Allow/disallow the process to change its mode to kernel
[NO]DETACH	Allow/disallow the process to create detached processes
[NO]DIAGNOSE	Allow/disallow the process to issue diagnostic I/O requests
[NO]EXQUOTA	Allow/disallow the process to exceed its quotas
[NO]GROUP	Allow/disallow the process to control other processes in the same group

Privilege	**Meaning**
[NO]GRPNAM	Allow/disallow the process to place names in the group logical name table
[NO]LOG_IO	Allow/disallow the process to issue logical I/O requests to a device
[NO]MOUNT	Allow/disallow the process to issue a mount volume QIO request
[NO]NETMBX	Allow/disallow the process to create a network device
[NO]OPER	Allow/disallow the process to perform operator functions
[NO]PFNMAP	Allow/disallow the process to create or delete sections mapped by page frame number
[NO]PHY_IO	Allow/disallow the process to issue physical I/O requests to a device
[NO]PRMCEB	Allow/disallow the process to create permanent common event flag clusters
[NO]PRMGBL	Allow/disallow the process to create permanent global sections
[NO]PRMMBX	Allow/disallow the process to create permanent mailboxes
[NO]PSWAPM	Allow/disallow the process to alter its swap mode
[NO]SAME	Allow/disallow the process to have the same privileges as the current process
[NO]SETPRV	Allow/disallow the process to give higher privileges to other processes
[NO]SHMEM	Allow/disallow the process to create or delete data structures in shared memory
[NO]SYSGBL	Allow/disallow the process to create system global sections
[NO]SYSNAM	Allow/disallow the process to place names in the system logical name table
[NO]SYSPRV	Allow/disallow access to files and other resources as if the user has a system UIC
[NO]TMPMBX	Allow/disallow the process to create temporary mailboxes
[NO]VOLPRO	Allow/disallow the process to override volume protection
[NO]WORLD	Allow/disallow the process to control all other processes in the system

If you do not specify /PRIVILEGES, the created process has the same privileges as your current process. If you specify /PRIVILEGES=NOSAME, the created process has no privileges.

/PROCESS_NAME=process-name

Defines a 1- through 15-alphanumeric character name for the created process. The process name is implicitly qualified by the group number of the process's UIC.

If you do not specify a process name, the created process has a null name, by default.

/QUEUE_LIMIT=quota

Specifies the maximum number of timer queue entries that the created process can have outstanding at any one time. This includes timer requests and scheduled wakeup requests.

If you do not specify a timer queue entry quota, the system uses the default value established at system generation time. A process does not require any timer queue quota in order to execute. A value of 8 is typical.

This quota is pooled.

/RESOURCE_WAIT
/NORESOURCE_WAIT

Enables or disables resource wait mode for the created process. By default, the system places a process in a wait state when a resource required for a particular function is not available.

If you specify /NORESOURCE_WAIT, the process will receive an error status code when a resource is unavailable. /RESOURCE_WAIT is the default.

/SCHEDULE=absolute-time

Requests that the created process be placed in hibernation and awakened at a specific time of day.

Specify the absolute time value according to the rules for entering absolute time values given in Section 5.8.

/SERVICE_FAILURE
/NOSERVICE_FAILURE

Enables or disables system service failure exception mode for the created process. By default, if an error occurs when a process calls a system service, a status code indicating an error is returned.

If you specify /SERVICE_FAILURE, the process will encounter an exception condition if an error occurs during a system service request. The default is /NOSERVICE_FAILURE.

/SUBPROCESS_LIMIT=quota

Specifies the maximum number of subprocesses that the created process is allowed to create.

If you do not specify a subprocess limit, the system uses the default value established at system generation time. A process does not require any subprocess quota in order to execute. A value of 8 is typical.

This quota is pooled.

/SWAPPING
/NOSWAPPING

Enables or disables process swap mode for the created process. By default, a process is swapped from the balance set in physical memory to allow other processes to execute.

If you specify /NOSWAPPING, the process is not swapped out of the balance set when it is in a wait state. /SWAPPING is the default. You must have the user privilege to disable process swapping (PSWAPM) to specify /NOSWAPPING for a process you create.

/TIME_LIMIT=limit

Specifies the maximum amount of CPU time allocated to the created process, in delta time, where the resolution is to ten milliseconds. When the time expires, the process is deleted. The default value is established at system generation time. A CPU time limit of 0 indicates that CPU time is not restricted; this is a typical value.

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 5.8.

This quota is deductible.

/UIC=uic

Specifies that the created process is to be a detached process.

The specified uic defines a user identification code for the created process, in the format:

[g,m]

g	is an octal number in range of 0 through 377 representing the group number of the process
m	is an octal number in the range of 0 through 377 representing the individual member number of the process

The square brackets ([ ]) are required in the UIC specification.

/WORKING_SET=default

Specifies the default working set size for the created process; that is, the number of pages in the working set for the image to be executed.

If you do not specify a default working set size, the system uses the default value 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 working set quota (specified with the /MAXIMUM_WORKING_SET qualifier). A value of 200 pages is typical.

This quota is nondeductible.

Examples

1.
```
$RUN/PROCESS_NAME=SUBA   SCANLINE
%RUN-S-PROC_ID, identification of created process is 00010044
```

   The RUN command creates a subprocess named SUBA to execute the image SCANLINE. The system gives the subprocess an identification number of 00010044

2.
```
$RUN/INTERVAL=1:40/PROCESS_NAME=STAT   STATCHK
%RUN-S-PROC_ID, identification of created process is 00050023
    .
    .
    .
$CANCEL STAT
```

   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 /DELAY nor /SCHEDULE 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.

3.
```
$RUN/PROCESS_NAME=LYRA  LYRA-
$/OUTPUT=_TTB3: -
$/ERROR=_TTB3:
%RUN-S-PROC_ID, identification of created process is 000A002F
```

   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.

4.
```
$RUN/UIC=[100,4]/PRIVILEGES=(SAME,NOPSWAPM) -
$/NORESOURCE_WAIT   OVERSEER
%RUN-S-PROC_ID, identification of created process is 0001002C
```

   The RUN command creates a detached process to execute under the UIC [100,4]. It executes the image OVERSEER.EXE. 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.

5.
```
$ASSIGN/GROUP  [MALCOLM.TESTFILES]   TEST
$RUN/PROCESS=SUB  WATCH -
$/INPUT=TEST:OUT1 -
$/OUTPUT='F$LOGICAL("SYS$OUTPUT")
%RUN-S-PROC_ID, identification of created process is 0001002E
```

   The ASSIGN 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.

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


Options

CARD_READER
[NO]CONTROL_Y
DEFAULT
HOST
MAGTAPE
MESSAGE
[NO]ON
PASSWORD
PROCESS
PROTECTION
QUEUE/ENTRY
RMS_DEFAULT
TERMINAL
[NO]VERIFY
WORKING_SET
```

Prompts

What: option

Description

The SET command options listed above are described individually in this manual. Table 4 lists all the SET command options, including those that are generally reserved for use by system operators and managers.

Table 4
SET Command Options

Option[1]	Function
ACCOUNTING[1]	Initializes the accounting log file
CARD_READER	Defines the default ASCII translation mode for a card reader
[NO]CONTROL_Y	Disables/enables interrupts caused by CTRL/Y
DEFAULT	Establishes a device and/or directory as the current default for file specifications
DEVICE[1]	Defines device characteristics
HOST	Connects the user's terminal through the current host processor to another remote VAX-11 processor
LOGINS[1]	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
PASSWORD	Allows users to change their own passwords
PRINTER[1]	Defines characteristics of a printer
PROCESS	Defines execution characteristics of the current process
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 during the job
QUEUE/ENTRY	Changes the attributes associated with one or more entries in a printer or batch job queue

1. Indicates that this command is described in the VAX/VMS Operator's Guide.

(continued on next page)

Table 4 (Cont.)
SET Command Options

Option[1]	Function
RMS_DEFAULT	Provides default multi-block and multi-buffer count values to be used by RMS for file operations
TERMINAL	Defines operational characteristics of a terminal
TIME[1]	Resets the system clock to the specified value
UIC[1]	Changes the UIC of the current process
[NO]VERIFY	Controls whether the command interpreter displays lines in command procedures as it executes them
WORKING_SET	Changes the current working set limit or quota

1. Indicates that this command is described in the VAX/VMS Operator's Guide.

SET CARD_READER

Defines the default translation mode for cards read from a card reader. All subsequent input read from the specified card reader will be converted using the specified mode.

Format

```
SET CARD_READER    device-name[:]

Command Qualifiers        Defaults

/026                      None.
/029
```

Prompts

Device: device-name[:]

Command Parameters

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 booted, 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.

Command Qualifiers

/026

Indicates that the cards were punched on an 026 punch.

/029

Indicates that the cards were punched on an 029 punch.

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 into the card reader CRA0 into the file CARDS.DAT in the directory MALCOLM.DATAFILES.

SET CONTROL_Y

Controls whether the command interpreter receives control when CTRL/Y is pressed.

Format

```
SET [NO]CONTROL_Y
```

Command Qualifiers	Defaults
None.	None.

Prompts

None.

Command Parameters

None.

Description

The CTRL/Y function key provides a general-purpose escape; it can be used at any time during an interactive terminal session to interrupt the current command, command procedure, or program image.

The SET NOCONTROL_Y command is provided for use in special applications; 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 CTRL/Y function key has the same effect as a CTRL/U function followed by a carriage return.

The effect of SET NOCONTROL_Y also applies to the CTRL/C function for all commands and programs that do not have special action routines to respond to CTRL/C.

Examples

1.
```
$ SET NOCONTROL_Y
```

   After this command, the CTRL/Y function is disabled.

SET DEFAULT

Changes the default device and/or directory name for the current process. The new default is applied to all subsequent file specifications that do not explicitly give 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[:]

Command Qualifiers                     Defaults

None.                                  None.
```

Prompts

Device: device-name[:]

Command Parameters

device-name[:]

Specifies a device and/or directory name to be used as the default device in file specifications.

If you specify a physical device name, terminate the device name with a colon. If you specify a directory name, you must enclose it in brackets ([ ] or < >).

The SET DEFAULT command performs logical name translation on the entire string specified, not on the left-most portion of the device name specified, as is the usual case. The translation is not recursive.

You may specify the minus sign (-) as a directory searching wild card character in the directory specification. See Section 2.1.6.3.

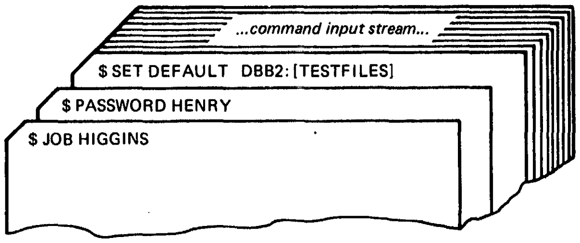
Examples

1.
```
$ SET DEFAULT [CARPENTER]
$ COPY A.* B.*
```

The SET DEFAULT command changes the default directory to CARPENTER. The default disk device does not change. The directory name CARPENTER is assumed to be the default directory for subsequent file searches, as in the COPY command shown.

2. `$ SET DEFAULT DBA2:`

   This command changes the default disk device to DBA2. The default directory name does not change.

3. 

   A batch user submits a job in the card reader. The first command in the batch job is a SET DEFAULT command; all file specifications will default to the directory TESTFILES on the disk DBB2.

4.
```
$ SAVEDEF := 'F$DIRECTORY()
$ SET DEFAULT  [122001.MALCOLM.TESTFILES]
    .
    .
    .

$ SET DEFAULT  'SAVEDEF'
```

   This command procedure uses the F$DIRECTORY lexical function to save the current default directory in the symbol named SAVEDEF. The SET DEFAULT command changes the default directory; later the symbol SAVEDEF is used to restore the original default.

SET HOST

Connects your terminal (through the current host processor) to another processor, called the remote processor. The remote processor must be a VAX-11. Both processors must be running DECnet-VAX.[1]

Format

```
SET HOST  node-name
```

Command Qualifiers	Defaults
None.	None.

Prompts

Node: node-name

Command Parameters

node-name

Specifies the node name for the remote processor.

Description

Use the SET HOST command to connect to another VAX-11 processor on a network. (Use the SHOW NETWORK command to obtain the names of nodes accessible to your node.) Once the connection is made, the remote processor prompts for the 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, and so on.

Use the LOGOUT command to log off the last processor you have logged in on. If you have connected to and logged in on more than one processor, the LOGOUT command leaves you logged in on the next-to-last processor.

For example, if your local node is GALAXY, you can use SET HOST STAR to connect to the node STAR; you can then use SET HOST ORION to connect (still through GALAXY and STAR) to the node ORION.

1. Available under separate license.

If you then use the LOGOUT command, you have logged off (and disconnected from) the processor at node ORION, but you are still logged in on (and connected to) the processor at STAR. A second LOGOUT command logs you off STAR, and disconnects you from it. A third LOGOUT command logs you off the local processor, GALAXY.

You can abort operations and return directly to the original host processor, if necessary. Press CTRL/Y at least two times in rapid succession. You will be prompted:

```
Are you repeating ^Y to abort the remote session?
```

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 issuing a series of LOGOUT commands or when part of the network becomes disconnected and you want to return to the host.

Examples

1.
```
$ SET HOST STAR
Username:
```

   This SET HOST command connects the user terminal to the processor at the network node named STAR. The remote processor then prompts for user name and password. Use the normal logging-in procedure to log in on the remote processor.

SET MAGTAPE

Defines the default characteristics associated with a specific magnetic tape device for subsequent file operations. The SET MAGTAPE command is valid for tape devices that do not currently have volumes mounted on them, or on which foreign volumes are mounted.

Format

```
    SET MAGTAPE     device-name[:]

    Command Qualifiers                        Defaults

    /DENSITY=density                          None.
    /REWIND
    /UNLOAD
```

Prompts

Device: device-name[:]

Command Parameters

device-name[:]

Specifies the name of the tape device for which the characteristics are to be set.

The device must not be currently allocated to any other user.

Command Qualifiers

/DENSITY=density

Specifies the default density, in bpi (bits per inch), for all write operations on the 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 tape drive.

/REWIND

Requests that the volume on the specified device be rewound to the beginning of the tape.

/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 mounts a foreign tape on the device MTB1. The SET MAGTAPE command defines the density for writing the tape at 800 bpi.

SET MESSAGE

Permits you to specify the display format of messages or to override or supplement the system messages.

Format

```
SET MESSAGE   [file-spec]

Command Qualifier          Defaults

/DELETE
/[NO]FACILITY              /FACILITY
/[NO]IDENTIFICATION        /IDENTIFICATION
/[NO]SEVERITY              /SEVERITY
/[NO]TEXT                  /TEXT
```

Prompts

File: file-spec

Command Parameters

file-spec

Specifies an optional message file. If the file specification does not contain a file type, the default type is EXE.

No wild card characters are allowed in the file specification.

Description

The SET MESSAGE command allows you to specify which message components VMS displays. The message components described in Section 1.5 can be summarized as:

```
%FACILITY-L-IDENT, text
```

When a process is created initially, the default is to display all four message components. If you want your message displays to omit the FACILITY component, you include the /NOFACILITY qualifier with the SET MESSAGE command. Similarly, the /NOIDENTIFICATION qualifier suppresses the IDENT portion shown above.

You may want to consider the SET MESSAGE command for inclusion in your login command file to select specific portions of the messages for your process.

The SET MESSAGE command also allows you to specify message definitions to override or supplement the system messages for your process. Note that the new definitions only affect your process.

The message definitions you specify must result from a successful compilation with the MESSAGE command. For full details of the use of the Message Utility to create your own messages, see the VAX-11 Utilities Reference Manual.

Whenever any software detects an error situation and invokes the $GETMSG system service, the message files are searched in the following order: image message sections first, then process permanent message files, and then the system message file. Thus, when you specify a message file with the SET MESSAGE command, you can introduce messages earlier in the searching order and either override or supplement the system messages.

If a process permanent message file is already in effect when you specify the SET MESSAGE command with a file specification, then the old file is removed and the new file added.

Command Qualifiers

/DELETE

Removes the currently selected process message file from your process. You should not include a file specification when you issue the /DELETE qualifier.

/FACILITY
/NOFACILITY

Controls whether the facility name prefix is displayed for all messages that occur for your process.

/IDENTIFICATION
/NOIDENTIFICATION

Controls whether the identification prefix (an abbreviation that identifies the message) is included in all messages that occur for your process.

/SEVERITY
/NOSEVERITY

Controls whether the severity level is displayed for all messages that occur for your process.

/TEXT
/NOTEXT

Controls whether the message text is displayed for all messages that occur for your process.

Examples

1.

```
$ 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 issued, error messages include all components. However, the SET MESSAGE command establishes that the IDENT portion (the abbreviation for the message text) is omitted on 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.

2.

```
$ SET MESSAGE NEWMSG
```

The SET MESSAGE command specifies that the message text in NEWMSG.EXE supplements the existing system messages.

Controls whether the command interpreter performs error checking following the execution of commands in command procedures.

Format

```
SET [NO]ON
```

Command Qualifiers	Defaults
None.	None.

Prompts

None.

Command 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.

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 value.

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.

For additional information on error handling in command procedures, see the VAX/VMS Guide to Using Command Procedures.

Examples

1.
```
$ SET NOON
$ DELETE *.SAV;*
$ SET ON
$ COPY  *.OBJ  *.SAV
```

This command procedure routinely copies all object modules into new files with file types of SAV. The DELETE command deletes all existing files with that file type, if any. The SET NOON command ensures that the procedure will continue execution if there are not currently any files with that file type. Following the DELETE command, the SET ON command restores error checking. Then, the COPY command makes copies of all existing files with file types of OBJ.

SET PASSWORD

Allows users to change their own passwords.

Format

```
SET PASSWORD

Command Qualifiers                        Defaults

None.                                     None.
```

Prompts

Old password:	old password
New password:	new password
Verification:	new password

Command Parameters

None.

Description

To maintain secrecy, users may need to change their passwords from time to time. The SET PASSWORD command offers a means of doing this. However, the system manager can control which users have the right to change their passwords.

Passwords may contain from 1 through 31 characters. The valid characters are:

A through Z
a through z
0 through 9
$ (dollar sign)
_ (underscore)

Note that all lowercase characters are converted to uppercase before the password is encrypted.

When the old and new passwords are entered, the user input is not echoed (to help ensure secrecy). To protect against typing errors that are not seen when entering the new password, you must enter the desired new password twice.

If an error occurs, the password remains unchanged.

The following guidelines are recommended to minimize the chances that passwords can be discovered by the trial-and-error method or by exhaustive search:

- Passwords should be at least six characters long
- Names or words that are readily associated with any user should be avoided
- Change passwords at least once every month

Examples

1.
```
$ SET PASSWORD
Old password:
New password:
Verification:
```

In response to the SET PASSWORD command, the system inquires for the old password, then for the new password. Then the system asks for the new password again for verification purposes. If the user is authorized to change this account's password, the old password is given correctly, and the new password is given identically twice, the password is changed. Otherwise, an error message appears and the password remains unchanged.

SET PROCESS

Changes execution characteristics associated with the current process for the current terminal session or job.

Format

```
SET PROCESS

Command Qualifiers              Defaults

/PRIVILEGES=(privilege[,...])   None.
/[NO]RESOURCE_WAIT
/[NO]SWAPPING
```

Prompts

None.

Command Parameters

None.

Command Qualifiers

/PRIVILEGES=(privilege[,...])

Enables or disables the user privileges for the current process. The privilege consists of one or more of the keywords listed below. Where multiple keywords are involved, they should be separated by commas and enclosed in parentheses.

You must have the SETPRV user privilege to enable a privilege that you are not authorized to possess. Otherwise, the privilege is not enabled, and no warning message is issued. Use the SHOW PROCESS/PRIVILEGES command to determine what privileges are currently enabled.

Privilege	**Meaning**
[NO]ACNT	Allow/disallow the process to create processes for which no accounting messages are written
[NO]ALLSPOOL	Allow/disallow the process to allocate spooled devices
[NO]ALTPRI	Allow/disallow the process to set priority values
[NO]BUGCHK	Allow/disallow the process to make bug check error log entries

Privilege	**Meaning**
[NO]BYPASS	Allow/disallow the process to bypass UIC protection
[NO]CMEXEC	Allow/disallow the process to change its mode to executive
[NO]CMKRNL	Allow/disallow the process to change its mode to kernel
[NO]DETACH	Allow/disallow the process to create detached processes
[NO]DIAGNOSE	Allow/disallow the process to issue diagnostic I/O requests
[NO]EXQUOTA	Allow/disallow the process to exceed its quotas
[NO]GROUP	Allow/disallow the process to control other processes in the same group
[NO]GRPNAM	Allow/disallow the process to place names in the group logical name table
[NO]LOG_IO	Allow/disallow the process to issue logical I/O requests to a device
[NO]MOUNT	Allow/disallow the process to issue a mount volume QIO request
[NO]NETMBX	Allow/disallow the process to create a network device
[NO]OPER	Allow/disallow the process to perform operator functions
[NO]PFNMAP	Allow/disallow the process to create or delete sections mapped by page frame number
[NO]PHY_IO	Allow/disallow the process to issue physical I/O requests to a device
[NO]PRMCEB	Allow/disallow the process to create permanent common event flag clusters
[NO]PRMGBL	Allow/disallow the process to create permanent global sections
[NO]PRMMBX	Allow/disallow the process to create permanent mailboxes
[NO]PSWAPM	Allow/disallow the process to alter its swap mode
[NO]SAME	Allow/disallow the process to have the same privileges as the current process
[NO]SETPRV	Allow/disallow the process to give higher privileges to other processes
[NO]SHMEM	Allow/disallow the process to create or delete data structures in shared memory

Privilege	Meaning
[NO]SYSGBL	Allow/disallow the process to create system global sections
[NO]SYSNAM	Allow/disallow the process to place names in the system logical name table
[NO]SYSPRV	Allow/disallow access to files and other resources as if the user has a system UIC
[NO]TMPMBX	Allow/disallow the process to create temporary mailboxes
[NO]VOLPRO	Allow/disallow the process to override volume protection
[NO]WORLD	Allow/disallow the process to control all other processes in the system

/RESOURCE_WAIT
/NORESOURCE_WAIT

Enables or disables resource wait mode for the current process. By default, the system places a process in a wait state when a resource required for a particular function is not immediately available.

If you specify /NORESOURCE_WAIT, the process will receive 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
Buffered I/O byte count (buffer space) quota

The default mode is /RESOURCE_WAIT.

/SWAPPING
/NOSWAPPING

Enables or disables process swap mode for the current process. 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. You must have the user privilege PSWAPM to disable swapping for your process.

Examples

1.
```
$ SET PROCESS/NORESOURCE_WAIT
```

   The SET PROCESS command disables resource wait mode for the current process.

SET PROCESS/PRIORITY

Changes the priority associated with a process for the current terminal session or job. The /PRIORITY qualifier is required.

Format

```
    SET PROCESS/PRIORITY=n     [process-name]


        Additional
    Command Qualifiers             Defaults

    /IDENTIFICATION=process-id     None.
```

Prompts

None.

Command Parameters

n

Specifies the new base priority for the requested process. A priority must be in the range of 0 through 31, where priorities 0 through 15 are reserved for normal processes and priorities 16 through 31 are reserved for real-time processes.

process-name

Specifies the 1- through 15-alphanumeric character-string name of a process whose priority is to be changed. The specified process must have the same group number in its user identification code as the current process.

If you specify the /IDENTIFICATION qualifier, the process-name parameter is ignored. If you specify neither the process-name parameter nor the /IDENTIFICATION qualifier, the priority for your current process is changed.

Description

The user privilege ALTPRI is required to increase the priority for any process to a value higher than the base priority of the current process. If you do not have the ALTPRI privilege, the value you specify is compared with your current base priority and the lower value is always used.

You can increase or decrease a priority. Note that if you decrease your own base priority and you do not have ALTPRI privilege, you cannot restore its original value.

The GROUP and WORLD privileges are required to control other processes in the same group or other processes in the system, respectively.

Command Qualifiers

/IDENTIFICATION=process-id

Specifies the process identification the system assigned to the process when the process was created. When you specify a process identification, you can omit leading zeros.

Examples

1.
```
$ RUN/PROCESS_NAME=TESTER  CALC
%RUN-S-PROC_ID, identification of created process is 0005002F
$ SET PROCESS/PRIORITY=10  TESTER
```

The RUN command creates a subprocess and gives it the name TESTER. Subsequently, the SET PROCESS/PRIORITY command assigns the subprocess a priority of 10.

SET PROTECTION

Establishes the protection to be applied to a particular file or a group of files. The protection for a file limits the type of access available to other system users.

Format

```
SET PROTECTION[=code]      file-spec[,...]

Command Qualifiers         Defaults

/[NO]CONFIRM               /NOCONFIRM
/[NO]LOG                   /NOLOG

File Qualifiers            Defaults

/PROTECTION=code           None.
```

Prompts

File: file-spec[,...]

Command Parameters

code

Defines the protection to be applied to the file(s) specified, if any.

The format for specifying the code is described in Section 5.10.

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 for only the highest existing version of the file.

You can specify wild card characters in the directory, file name, file type, and version fields. See Section 2.1.6.

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, allow you to define the protection for individual files.

If you omit both the code and the /PROTECTION file qualifier, your current default protection (established by the SET PROTECTION/DEFAULT command) is applied to the file.

Command Qualifiers

/CONFIRM
/NOCONFIRM

Controls whether the SET PROTECTION command displays the file specification of each file before applying the new protection and requests you to confirm whether or not the file's protection actually should be changed. If you specify /CONFIRM, you must respond to a prompt with a Y (YES) or a T (TRUE) followed by a carriage return, before the SET PROTECTION command changes the file protection. If you enter anything else, such as N or NO, the requested file protection is not applied.

By default, the SET PROTECTION command does not request confirmation of the files it is affecting.

/LOG
/NOLOG

Controls whether the SET PROTECTION command displays the file specification of each file after it sets the protection.

By default, the SET PROTECTION command does not display the names of files after it sets their protection.

File Qualifiers

/PROTECTION=code

Defines the protection code to be applied to the associated file specification. Use this qualifier to assign different protection codes to several files in a single command.

If you specify the code parameter in addition to qualifying file specifications with the /PROTECTION qualifier, the attributes specified with the code parameter are applied first, then any attributes specified with the qualifier override them.

Specify the code in the format described in Section 5.10.

Examples

1.
```
$ DELETE INCOME.DAT;3
%DELETE-W-FILNOTDEL, error deleting _DB1:[MALCOLM]INCOME.DAT;3
-RMS-E-PRV, privilege violation (operating system denies access)
$ SET PROTECTION=OWNER:D INCOME.DAT;3
$ DELETE INCOME.DAT;3
```

   The file INCOME.DAT;3 had been protected against deletion. This SET PROTECTION command changes the owner's access to delete-only for the file INCOME.DAT;3. Now the file can be deleted.

2.
```
$ SET PROTECTION -
$_PAYROLL.LIS/PROTECTION=(SYSTEM:R,OWNER:RWED,GROUP:RW),-
$_PAYROLL.OUT/PROTECTION=(SYSTEM:RWED,GROUP:RWED)
```

   This SET PROTECTION command changes the protection codes applied to two files. To the file PAYROLL.LIS, it gives the system read-only access, the owner read, write, execute, and delete access, and users in the owner's group read/write access. To the file PAYROLL.OUT, it gives the system and group all types of access, and does not change the current access for owner and world.

3.
```
$ SET PROTECTION A.DAT,B.DAT/PROTECTION=OWNER:RWED,C.DAT
```

   The SET PROTECTION command specifies that the file A.DAT should receive the default protection established for your files. The existing protection for the file B.DAT is overridden for the owner category only, 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. Thus, like A.DAT, C.DAT receives your default protection.

   Since no version numbers are specified in this example, 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 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.

Establishes the default protection for all files subsequently created during the terminal session or batch job. The protection for a file limits the type of access available to other system users. The /DEFAULT qualifier is required.

Format

```
SET PROTECTION[=code]/DEFAULT
```

Additional Command Qualifiers	Defaults
None.	None.

Prompts

None.

Command Parameters

code

Defines the protection to be applied to all files subsequently created in cases where the specific protection is not specified by the SET PROTECTION or CREATE commands.

The format for specifying the protection code is described in Section 5.10.

If you fail to specify a protection code, the current default protection remains unchanged.

Examples

1.
```
$ SET PROTECTION=(GROUP:RWED,WORLD:R)/DEFAULT
```

This SET PROTECTION/DEFAULT command sets the default protection applied to all files subsequently created to allow other users in the same group unlimited access, and all users read access. Default protection for system and owner are not changed.

SET QUEUE/ENTRY

Changes the current status or attributes of a file that is queued for printing or for batch job execution but not yet processed by the system. The /ENTRY qualifier is required.

Format

```
    SET QUEUE/ENTRY=job-number     [queue-name[:]]


        Additional
    Command Qualifiers                       Defaults

    /AFTER=absolute-time                     None.
    /CHARACTERISTICS=(c[,...])
    /CPUTIME=n
    /FORMS=type
    /HOLD
    /JOB_COUNT=n
    /[NO]LOWERCASE
    /NAME=job-name
    /PRIORITY=n
    /RELEASE
    /WSDEFAULT=n
    /WSQUOTA=n
```

Prompts

None.

Command Parameters

job-number

Specifies the job number of the job you want to change.

queue-name[:]

Specifies the name of the queue in which the specified file is entered. No logical name translation is performed on the specified queue name.

If you do not specify a queue name, the system assumes the default name of SYS$PRINT.

Description

The system assigns a unique entry number, called a job number, to each queued printer or batch job in the system. The PRINT and SUBMIT commands display the job number when they successfully queue a job for processing. Use this job number to specify the entries you want to change.

Additional Command Qualifiers

/AFTER=absolute-time

Requests that the specified job be held until a specific time, then released for printing. If the specified time has already passed, the file is released immediately.

Specify the time value according to the rules for entering absolute times given in Section 5.8.

/CHARACTERISTICS=(c[,...])

Replaces characteristics for the specified job. This qualifier overrides the characteristics specified on the PRINT command.

If you specify more than one characteristic, separate them by commas and enclose the list in parentheses.

Printer characteristics are described in the VAX/VMS System Manager's Guide.

/CPUTIME=n

Defines a CPU time limit for the batch job. You may specify a delta time (Section 5.8.2), the value 0, or the words NONE or INFINITE for n.

Use this qualifier to override the base queue value established by the system manager or the value authorized in your user authorization file, when you need less CPU time than authorized. 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 value or the limit specified on the queue. (However, you cannot request more time than permitted by the base limits or your user authorization file.)

/FORMS=n

Modifies the forms type for the specified job. This qualifier overrides the forms type specified or defaulted on the PRINT command.

Forms type specifications are described in the VAX/VMS System Manager's Guide.

/HOLD

Requests that the specified job(s) be placed in a hold status. Jobs in a hold status are not processed until you release them with the /RELEASE qualifier of the SET QUEUE/ENTRY command.

/JOB_COUNT=n

Specifies the number of copies of the job to print. This qualifier overrides the /JOB_COUNT qualifier specified or defaulted on the PRINT command.

/LOWERCASE
/NOLOWERCASE

Indicates whether the specified job(s) must be printed on a printer with lowercase letters.

/NAME=job-name

Defines a 1- through 8-alphanumeric character name string to identify the job, overriding the job name assigned to the job when it was queued.

/PRIORITY=n

Changes the priority of a job relative to other jobs that are currently queued. The priority, n, must be in the range of 0 through 31, where 0 is the lowest priority and 31 is the highest.

By default, jobs are assigned the same priority as your current process priority; you must have the operator user privilege (OPER) to set a priority value greater than the base priority of your current process.

/RELEASE

Releases a previously held job for processing.

/WSDEFAULT=n

Defines a working set default for the batch job. You may specify a positive integer in the range 1 through 65535 or the word NONE for n.

Use this qualifier to override the base queue value established by the system manager or the value authorized in your user authorization file, provided you want to impose a lower value. Specify 0 or NONE if you want the working set value defaulted to either your user authorization file value or the working set quota specified on the queue. However, you may not request a higher value than your default.

/WSQUOTA=n

Defines the maximum working set size for the batch job. This is the working set quota. You may specify a positive integer in the range 1 through 65535,.0, or the word NONE for n.

Use this qualifier to override the base queue value established by the system manager or the value authorized in your user authorization file, provided you want to impose a lower value. 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. However, you may not request a higher value than your default.

Examples

1.
```
$ PRINT/HOLD   MYFILE.DAT
  Job 112 entered on queue SYS$PRINT
    .
    .
    .

$ SET QUEUE/ENTRY=112/RELEASE/JOB_COUNT=3
```

The PRINT command requests that the file MYFILE.DAT be queued to the system printer, but placed in a hold status. The SET QUEUE/ENTRY command releases the file for printing and changes the number of copies of the job to three.

2.
```
$ SUBMIT  WEATHER
  Job 210 entered on queue SYS$BATCH
$ SUBMIT  CLIMATE
  Job 211 entered on queue SYS$BATCH
$ SET  QUEUE/ENTRY=211/HOLD/NAME=TEMP  SYS$BATCH
```

The two SUBMIT commands queue command procedures for batch processing. The system assigns them job numbers of 210 and 211, respectively. The SET QUEUE/ENTRY command places the second job in a hold state and changes the job name to TEMP, provided job 211 had not yet begun.

SET RMS_DEFAULT

Defines default values for the multiblock and multibuffer counts used by VAX-11 RMS for file operations. Defaults can be set for sequential, indexed-sequential, or relative files on a process-only or system-wide basis.

Format

```
    SET RMS_DEFAULT


    Command Qualifiers        Defaults

    /BLOCK_COUNT=count
    /BUFFER_COUNT=count
    /DISK
    /INDEXED                  /SEQUENTIAL
    /MAGTAPE
    /PROCESS                  /PROCESS
    /RELATIVE                 /SEQUENTIAL
    /SEQUENTIAL               /SEQUENTIAL
    /SYSTEM                   /PROCESS
    /UNIT_RECORD
```

Prompts

None.

Command Parameters

None.

Description

Multiblocking and multibuffering of file operations can enhance the speed of input/output operations with VAX-11 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 VAX-11 Record Management Services Reference Manual.

Command Qualifiers

/BLOCK_COUNT=count

Specifies a default multiblock count for file operations. The specified count, representing the number of blocks to be allocated for each I/O buffer, can be in the range of 1 through 127.

/BUFFER_COUNT=count

Specifies a default multibuffer count for file operations. The specified count, representing the number of buffers to be allocated, can be in the range of -128 through 127. A positive value indicates the specified number of buffers must be locked in the process's working set for the I/O operation. A negative value indicates that the specified number of buffers must be allocated but do not have to be locked.

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.

/DISK

Indicates that the specified default(s) are to be applied to file operations on disk devices. If /SEQUENTIAL or /RELATIVE is specified, /DISK is assumed.

/INDEXED

Indicates that the specified default(s) are to be applied to indexed file operations.

/MAGTAPE

Indicates that the specified default(s) are to be applied to operations on magnetic tape volumes.

/PROCESS

Indicates that the specified defaults are to be applied to file operations occurring within the current process.

/RELATIVE

Indicates that the specified defaults are to be applied to file operations on relative files.

/SEQUENTIAL

Indicates that the specified defaults are to be applied to all sequential file operations, including operations on disk, magnetic tape, and unit record devices.

/SEQUENTIAL is the default if neither /RELATIVE nor /INDEXED is specified.

/SYSTEM

Indicates that the specified defaults are to be applied to file operations by all processes.

The operator user privilege (OPER) is required to set the default for the system.

/UNIT_RECORD

Indicates that the specified default(s) are to be applied to file operations on unit record devices.

Examples

1.
```
$ SET RMS_DEFAULT/DISK/BLOCK_COUNT=16
```

   The SET RMS_DEFAULT command defines the default multiblock count for disk file input/output operations as 16 blocks. This default is defined only for the current process, and will be used for disk file operations in user programs that do not explicitly set the multiblock count.

2.
```
$ SET RMS_DEFAULT/BUFFER_COUNT=8/MAGTAPE
```

   The SET RMS_DEFAULT command defines the default multibuffer count for input/output operations on magnetic tapes as eight buffers.

SET TERMINAL

Changes the characteristics of a specified terminal.

Format

```
SET TERMINAL     [device-name[:]]
```

Command Qualifiers	Defaults
/[NO]BROADCAST	/BROADCAST
/CRFILL[=formula]	/CRFILL=0
/[NO]ECHO	/ECHO
/[NO]EIGHT_BIT	/NOEIGHT_BIT
/[NO]ESCAPE	/NOESCAPE
/[NO]FORM	
/FT1	
/FT2	
/FT3	
/FT4	
/FT5	
/FT6	
/FT7	
/FT8	
/[NO]FULLDUP	/HALFDUP
/[NO]HALFDUP	/HALFDUP
/[NO]HARDCOPY	
/[NO]HOLD_SCREEN	/NOHOLD_SCREEN
/[NO]HOSTSYNC	/NOHOSTSYNC
/[NO]INTERACTIVE	/INTERACTIVE
/LA36	
/LA120	
/LFFILL[=formula]	
/[NO]LOWERCASE	
/PAGE[=n]	/PAGE=0
/[NO]PARITY[=option]	/NOPARITY
/[NO]PASSALL	/NOPASSALL
/PERMANENT	
/[NO]READSYNC	/NOREADSYNC
/[NO]SCOPE	
/SPEED=rate	
/[NO]TAB	/NOTAB
/[NO]TTSYNC	/TTSYNC
/[NO]TYPE_AHEAD	/TYPE_AHEAD
/UNKNOWN	
/[NO]UPPERCASE	
/VT05	
/VT52	
/VT55	
/VT100	
/WIDTH=n	
/[NO]WRAP	/WRAP

Prompts

None.

Command Parameters

device-name[:]

Specifies the name of the terminal whose characteristics are to be changed.

If you do not specify a device name, the qualifiers change the characteristics of the current SYS$COMMAND device, if SYS$COMMAND is a terminal.

Description

The SET TERMINAL command allows you to modify specific terminal characteristics for a particular application, or to override system default characteristics. (These defaults are defined on an individual installation basis, based on the most common type of terminal in use.)

The following qualifiers modify more than one characteristic, based on the specific type of terminal:

/LA36
/LA120
/VT05
/VT52
/VT55
/VT100

In addition to the terminal type qualifiers above, there are the foreign terminal qualifiers, /FT1 through /FT8, and the /UNKNOWN qualifier. All these terminal type qualifiers are mutually exclusive.

The settings affected by each of these qualifiers are summarized in Table 5.

The terminal characteristics of local or remote are determined automatically by the terminal driver. These characteristics are not affected by the SET TERMINAL command. For example, when you successfully dial into a VAX-11, you establish your terminal as remote. When you hang up, the terminal characteristic is set back to local.

Command Qualifiers

/BROADCAST
/NOBROADCAST

Controls whether the terminal can receive messages broadcast by the system operator. By default, a terminal receives any messages the system operator or another privileged user sends.

Use /NOBROADCAST when you are using a terminal as a non-interactive terminal or when you do not want special output to be interrupted by messages.

/CRFILL[=formula]

Specifies whether the system must generate fill characters following a carriage return on the terminal.

The formula is a number in the range of 0 through 9 indicating the number of null fill characters required to ensure that the carriage return completes successfully before the next meaningful character is sent. You may need to use this qualifier if you are using a non-DIGITAL terminal or a video terminal. This qualifier prevents the system from sending out data before the terminal is ready to accept it.

The default is /CRFILL=0.

/ECHO
/NOECHO

Controls whether the terminal echoes, or displays, the input lines that it receives.

When /NOECHO is set, the terminal displays only data that a system or user application program writes to it.

/EIGHT_BIT
/NOEIGHT_BIT

Indicates whether the terminal uses an 8-bit ASCII character code.

The default is /NOEIGHT_BIT; the terminal interprets characters using 7-bit ASCII code.

/ESCAPE
/NOESCAPE

Indicates whether the terminal generates valid escape sequences that will be interpreted by an applications program controlling the terminal.

If you specify /ESCAPE, the terminal checks the escape sequences for syntax before passing them to the program. For information on escape sequences, see the VAX/VMS I/O User's Guide.

/FORM
/NOFORM

Controls whether the terminal driver translates form feed characters into line feeds or merely outputs the untranslated form feed character. Only a few terminals can accept form feeds (such as the LA120 and VT100). Thus, the default for all terminals except the LA120 is /NOFORM. See Table 5.

/FT1
/FT2
/FT3
/FT4
/FT5
/FT6
/FT7
/FT8

Permits up to eight different categories of terminals that are not supported by VAX/VMS to be identified as foreign terminals. Foreign terminals may be handled specially by user's software. See the VAX/VMS I/O User's Guide for instructions on how to obtain the terminal type from a user program.

When you establish the terminal type as foreign through one of the /FTn qualifiers, you do not change the default characteristics established for the terminal.

/FULLDUP
/NOFULLDUP

Specifies whether the terminal's mode of operation is full duplex or half duplex.

For a description of these modes of operation, see the terminal driver chapter in the VAX/VMS I/O User's Guide.

This qualifier is complementary to the /HALFDUP qualifier; that is, /FULLDUP is equivalent to /NOHALFDUP. The default is /NOFULLDUP.

/HALFDUP
/NOHALFDUP

Specifies whether the terminal's mode of operation is full duplex or half duplex.

For a description of these modes of operation, see the terminal driver chapter in the VAX/VMS I/O User's Guide.

This qualifier is complementary to the /FULLDUP qualifier; that is, /HALFDUP is equivalent to /NOFULLDUP. The default is /HALFDUP.

/HARDCOPY
/NOHARDCOPY

Indicates whether the terminal prints hardcopy output, as opposed to a video terminal. It also affects how the terminal interprets certain input keys. The /HARDCOPY qualifier establishes the terminal as a hardcopy device. Thus, the RUBOUT or DELETE key cannot accomplish backspace deletions. Instead, the text being replaced is surrounded in backslash characters (\).

This qualifier is complementary to the /SCOPE qualifier, that is, /HARDCOPY is equivalent to /NOSCOPE.

/HOLD_SCREEN
/NOHOLD_SCREEN

Enables and disables the operation of the SCROLL key on VT52, VT55, and VT100 (in VT52 mode) video terminals. Enabling the SCROLL key allows you to display a screenful of data at a time. The default is /NOHOLD_SCREEN.

If you specify /HOLD_SCREEN, the SET TERMINAL command also sets the /TTSYNC qualifier.

Once you have established the HOLD_SCREEN characteristic on your terminal, you may press the SHIFT and SCROLL keys simultaneously to display each screenful of output. You may also advance the screen output one line at a time by pressing just the SCROLL key.

/HOSTSYNC
/NOHOSTSYNC

Controls whether the system can synchronize the flow of input from the terminal.

When you specify the /HOSTSYNC qualifier, the system generates CTRL/S and CTRL/Q to enable or disable the reception of input. When the type-ahead buffer capacity is full, the system sends CTRL/S to temporarily stop input; when the buffer is empty, the system sends CTRL/Q so that more input can be entered. The size of the type-ahead buffer is established at the time of system generation.

/INTERACTIVE
/NOINTERACTIVE

Indicates that the terminal is in use as an interactive terminal.

This qualifier is complementary to the /PASSALL qualifier, that is, /INTERACTIVE is equivalent to /NOPASSALL.

/LA36

Indicates that the terminal is an LA36 terminal. When you specify this qualifier, default terminal characteristics for LA36 terminals are set. These settings are summarized in Table 5.

/LA120

Indicates that the terminal is an LA120 terminal. When you specify this qualifier, default terminal characteristics for LA120 terminals are set. These settings are summarized in Table 5.

/LFFILL[=formula]

Specifies whether the system must generate fill characters following a line feed on the terminal.

The formula is a number in the range of 0 through 9 indicating the number of null fill characters required to ensure that the line feed completes successfully before the next meaningful character is read. You may need to use this qualifier if you are using a non-DIGITAL terminal or a video terminal.

This qualifier prevents the system from sending out data before the terminal is ready to accept it.

The default is installation dependent. See Table 5.

/LOWERCASE
/NOLOWERCASE

Indicates whether the terminal has uppercase and lowercase characters.

If you specify /NOLOWERCASE all alphabetic characters are translated to uppercase. If you specify /LOWERCASE, lowercase characters are not converted to uppercase.

This qualifier is complementary to the /UPPERCASE qualifier, that is, /LOWERCASE is equivalent to /NOUPPERCASE.

/PAGE[=n]

Specifies the page length of the terminal. For hardcopy terminals, the page size, n, equals the number of print lines between perforations on the paper. When the terminal reads a form feed character, it advances the paper to the next perforation. A page size of 0 indicates that the terminal treats form feeds as if they were line feeds.

You can specify values of 0 through 255 for the page size. The default size is installation dependent. However, if you specify /PAGE without a value, the default value for n is 0.

/PARITY[=option]
/NOPARITY

Defines the parity for the terminal. If you have the LOG_IO user privilege, you can specify one of the following options:

EVEN
ODD

If you specify /PARITY and you do not specify the ODD option, the command assumes /PARITY=EVEN.

/PASSALL
/NOPASSALL

Controls whether the system interprets special characters or passes all data to an application program as 8-bit binary data.

A terminal operating with /PASSALL set does not expand tab characters to blanks, fill carriage return or line feed characters, or recognize control characters.

/PERMANENT

Controls whether the characteristics that are specified are established permanently or only for the current terminal session. By default the characteristics are only in effect for the current session.

The permanent characteristics are restored when the current user logs out. If you use the /PERMANENT qualifier to override the system default characteristics established at system generation time, remember that if the system is halted, the permanent characteristics revert to those defined at system generation.

You may want to use the /PERMANENT qualifier with the SET TERMINAL command in a system startup file to establish the characteristics for all terminals on the system.

You must have the LOG_IO or PHY_IO user privilege to specify the /PERMANENT qualifier.

/READSYNC
/NOREADSYNC

Controls whether the system solicits read data from a terminal using CTRL/S and terminates the read using CTRL/Q.

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 on special-purpose terminal lines where data synchronization is appropriate.

/SCOPE
/NOSCOPE

Indicates whether the terminal is a video terminal and, thus, how it reacts to certain keys. The /SCOPE qualifier establishes the terminal as a video terminal. Thus, when you press the DELETE key, the printing position is moved left one space and any character displayed in that position is erased.

This qualifier is complementary to the /HARDCOPY qualifier, that is, /SCOPE is equivalent to /NOHARDCOPY.

/SPEED=rate

Specifies the rate at which the terminal sends and receives data.

You can specify the rate as a single value to set the input and output baud rates to the same speed. To specify different baud rates for input and output, specify the rate in the format (n,m). The values n and m indicate the input and output baud rates, respectively. Not all interface devices support different input and output baud rates. Consult the appropriate hardware manual for details on a specific interface device.

The valid values for input and output baud rates are:

50	300	2400
75	600	3600
110	1200	4800
134	1800	7200
150	2000	9600

The default transmission rates are installation dependent.

/TAB
/NOTAB

Controls how the terminal handles tab characters. The default is /NOTAB, which expands all tab characters to blanks, assuming tab stops at 8-character intervals.

Use the /TAB qualifier when you do not want the system to convert tabs to blanks, but want the terminal to process the tab characters.

/TTSYNC
/NOTTSYNC

Controls whether the terminal synchronizes output by responding to CTRL/S and CTRL/Q.

The default is /TTSYNC; the system stops sending output when you press CTRL/S and resumes output when you press CTRL/Q.

/TYPE_AHEAD
/NOTYPE_AHEAD

Controls whether the terminal accepts unsolicited input (that is, input that you type when there is no outstanding read).

When you specify /NOTYPE_AHEAD, the terminal is dedicated, and will only accept input when a program or the system issues a read to it.

Use the /NOTYPE_AHEAD qualifier to ensure that a specific terminal remains dedicated to a particular application.

When you specify /TYPE_AHEAD, the amount of data that can be accepted is limited to the size of the type-ahead buffer. The size of the type-ahead buffer is established at system generation time.

/UNKNOWN

Indicates that the terminal is of an unknown terminal type. When you specify this qualifier, default terminal characteristics for terminals of an unknown type are set. For a summary of the settings, see Table 5.

/UPPERCASE
/NOUPPERCASE

Specifies whether or not the terminal should translate all lowercase letters to uppercase.

This qualifier is complementary to the /LOWERCASE qualifier, that is, /UPPERCASE is equivalent to /NOLOWERCASE.

/VT05

Indicates that the terminal is a VT05 terminal. When you specify this qualifier, default terminal characteristics for VT05 terminals are set. For a summary of the settings, see Table 4.

/VT52

Indicates that the terminal is a VT52 terminal. When you specify this qualifier, default terminal characteristics for VT52 terminals are set. These settings are summarized under "VT5x" in Table 5.

/VT55

Indicates that the terminal is a VT55 terminal. When you specify this qualifier, default terminal characteristics for VT55 terminals are set. These settings are summarized under "VT5x" in Table 5.

/VT100

Indicates that the terminal is a VT100 terminal. When you specify this qualifier, default terminal characteristics for VT100 terminals are set. For a summary of the settings, see Table 5.

/WIDTH=n

Specifies the number of characters on each input or output line. The width, n, must be in the range of 0 through 255.

If /WRAP is in effect, the terminal generates a carriage return/line feed when a line reaches the specified width.

/WRAP
/NOWRAP

Controls whether or not the terminal generates a carriage return/line feed when it reaches the end of the line. The end of a line is determined by the setting of the terminal width.

If you specify /NOWRAP and the terminal is accepting input, the terminal does not generate a carriage return/line feed when it reaches the end of a line, but continues to accept input at the last physical character position on the terminal line. If you specify /NOWRAP and the terminal is writing output, it continues to write characters out in the last position on the line.

If you specify /WRAP, the terminal generates a carriage return/line feed whenever the end of the line is reached.

Examples

1.
```
$ SET TERMINAL/VT52
```

This SET TERMINAL command establishes the current terminal as a VT52 terminal and sets the default characteristics for that terminal type.

2.
```
$ SET TERMINAL/WIDTH=132/PAGE=66/NOBROADCAST
$ TYPE MEMO.DOC
    .
    .
    .
$ SET TERMINAL/LA36
```

The first SET TERMINAL command indicates that the width of terminal lines is 132 characters and that the size of each page is 66 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.

Table 5
Default Characteristics for Terminals

Name Qualifier	UNKNOWN /UNKNOWN	FOREIGN /FT1-/FT8	LA36 /LA36	LA120 /LA120	VT05 /VT05	VT5x /VT52 /VT55	VT100 /VT100
BROADCAST	*	*	*	*	*	*	*
CRFILL	*	*	0	0	0	0	0
ECHO	*	*	yes	yes	yes	yes	yes
EIGHT_BIT	*	*	no	no	no	no	no
ESCAPE	*	*	*	*	no	*	*
FORM	*	*	no	yes	no	no	no
HALFDUP	*	*	yes	yes	yes	yes	yes
HOLD_SCREEN	*	*	no	no	no	*	*
HOSTSYNC	*	*	no	no	no	yes	yes
LFFILL	*	*	0	0	3	0	0
LOWERCASE	*	*	yes	yes	no	yes	yes
PAGE	*	*	8	8	20	24	24
PARITY	*	*	no	no	no	no	no
PASSALL	*	*	no	no	no	no	no
READSYNC	*	*	no	no	no	no	no
SPEED	*	*	*	*	*	*	*
TAB	*	*	no	no	no	yes	yes
TTSYNC	*	*	yes	yes	yes	yes	yes
TYPE_AHEAD	*	*	yes	yes	yes	yes	yes
WIDTH	*	*	132	132	72	80	80
WRAP	*	*	yes	yes	yes	yes	yes

* Indicates that the current setting is not changed by the qualifier.

SET VERIFY

Controls whether command lines in command procedures are displayed at the terminal or printed in a batch job log.

Format

```
SET [NO]VERIFY
```

Command Qualifiers	Defaults
None.	None.

Prompts

None.

Command Parameters

None.

Description

By default, when the system processes command procedures executed interactively, it does not display the command lines at the terminal. System responses and error messages are always displayed.

If you use the SET VERIFY command to override the default setting, the system displays all the lines in command procedures as it executes them. If any lines contain lexical functions or symbol names that are substituted before command execution, the command interpreter displays the line as it appears after symbol substitution.

When you change the verification setting, it remains in effect for all command procedures that you subsequently execute.

The default setting for a batch job is VERIFY; that is, all lines in the command procedure appear in the batch job listing.

For more information on how to use commands like this one in command procedures, consult the VAX/VMS Guide to Using Command Procedures.

Examples

1.

```
$ SET VERIFY
    .
    .
    .

$ SET NOVERIFY
$ EXIT
```

The verification setting is turned on for the execution of a command procedure. The system displays all the lines in the procedure, including command lines, as it reads them. At the end of the procedure, the SET NOVERIFY command restores the system default.

2.

```
$ VERIFYFLG = 'F$VERIFY()
$ SET NOVERIFY
    .
    .
    .

$ IF VERIFYFLG THEN SET VERIFY
```

This command procedure uses the lexical function F$VERIFY to save the current setting of verification in the symbol named VERIFYFLG (the function returns a value of 1 if verification is set on, a value of 0 if verification is set off). Then, the SET NOVERIFY command turns off verification. Subsequently, the IF command tests the value of VERIFYFLG; if true (1), then verification is restored; if false (0), then verification remains off.

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.

Format

SET WORKING_SET	
<u>Command Qualifiers</u>	<u>Defaults</u>
/LIMIT=n /QUOTA=n	None.

Prompts

None.

Command Parameters

None.

Description

A process's working set is the number of pages that are resident in physical memory when an image is executing in the process. 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.

Command Qualifiers

/LIMIT=n

Specifies the maximum number of pages that can be resident in the working set during image execution.

The value, n, must be greater than the minimum working set defined at system generation and it must be less than or equal to the authorized limit defined in the user authorization file.

If you specify a value greater than the authorized. limit, the command sets the working set limit at the maximum authorized value.

If you specify a value greater than the current quota, the quota value is also increased.

If you specify a limit equal to the /QUOTA value, automatic working set adjustment is disabled. See the <u>VAX/VMS System Manager's Guide</u> for a description of automatic working set adjustment.

/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 system service, which is described in the VAX/VMS System Services Reference Manual.

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=100  /Quota=200  Authorized Quota=200
$ SET WORKING_SET/QUOTA=500
  New Working Set /Limit=100  /Quota=200
```

   The SHOW WORKING_SET command displays the current limit, quota, and authorized quota. The SET WORKING_SET command attempts to set a quota limiting the maximum number of pages any image can request that is in excess of the authorized quota. Note from the response that the quota was not increased.

2.
```
$ SET WORKING_SET/LIMIT=200
```

   The SET_WORKING SET command sets both the working set size and the quota allowed to any image in the process to 200.

Displays information about the current status of the process, the system, or devices in the system.

Format

```
SHOW    option

Options

[DAY]TIME
DEFAULT
DEVICES
LOGICAL
MAGTAPE
NETWORK
PRINTER
PROCESS
PROTECTION
QUEUE
QUOTA
RMS_DEFAULT
STATUS
SYMBOL
SYSTEM
TERMINAL
TRANSLATION
WORKING_SET
```

Prompts

What: option

Description

The SHOW command options are summarized in Table 6. Each SHOW command option and the format of the information it displays is described separately following Table 6.

Table 6
SHOW Command Options

Option	Displays
[DAY]TIME	The current date and time
DEFAULT	The current default device and directory
DEVICES	The status of devices in the system
LOGICAL	Current logical name assignments
MAGTAPE	The status and characteristics of a specific magnetic tape device
NETWORK	The availability of network nodes, including the current node
PRINTER	Default characteristics of a line printer
PROCESS	Attributes of the current process, including privileges, resource quotas, memory usage, priority, and accounting information
PROTECTION	The current default protection applied to files
QUEUE	Printer or batch jobs that have been queued but have not completed
QUOTA	The current disk quota authorized for and used by a specific user on a specific disk
RMS_DEFAULT	The current default multi-block and multi-buffer 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
TRANSLATION	The result of translating a logical name
WORKING_SET	The current working set size limit and quota

SHOW DAYTIME

Displays the current date and time in the default output stream.

Format

```
SHOW [DAY]TIME

Command Qualifiers                     Defaults

None.                                  None.
```

Prompts

None.

Command Parameters

None.

Examples

1.
```
$ $HOW DAYTIME
  18-JAN-1977 00:03:45
```

The SHOW DAYTIME command requests a display of the current date and time.

SHOW DEFAULT

Displays the current default device and directory name. These defaults are applied whenever you omit a device and/or directory name from a file specification.

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.

Format

SHOW DEFAULT

Command Qualifiers	Defaults
None.	None.

Prompts

None.

Command Parameters

None.

Examples

1.
```
$ SHOW DEFAULT
  DBA1:[ALPHA]
$ SET DEFAULT DBA2:[HIGGINS.SOURCES]
$ SHOW DEFAULT
  DBA2:[HIGGINS.SOURCES]
```

   The SHOW DEFAULT command requests a display of the current default device and directory. The SET DEFAULT command changes these defaults, and the next SHOW DEFAULT command displays that the defaults have in fact been changed.

2.
```
$ ASSIGN DBA3:  SYS$DISK
$ SHOW DEFAULT
  DBA3:[HIGGINS2]
```

   The ASSIGN command changes the equivalence name for the logical name SYS$DISK. This also changes the device name default, as the response from the SHOW DEFAULT command indicates.

Displays the status of all devices in the system, the status of a particular device, or lists the devices that currently have volumes mounted on them and/or are allocated to processes.

Format

```
SHOW DEVICES    [device-name[:]]

Command Qualifiers        Defaults

/ALLOCATED
/BRIEF                    /BRIEF
/FULL                     /BRIEF
/MOUNTED
```

Prompts

None.

Command Parameters

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 non-specified 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 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 specify 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 in the system. To obtain information about a specific device or generic class of devices, specify a device name.

Use the /ALLOCATED or /MOUNTED qualifier for a list of devices that are currently allocated to processes or mounted, respectively.

Command Qualifiers

/ALLOCATED

Requests a display of 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

Requests a brief display of information about the device(s) specified.

/FULL

Requests a complete listing of information about the device(s).

/MOUNTED

Requests a display of 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 devices of that type that currently have volumes mounted are displayed.

Examples

1.
```
$ SHOW DEVICES
List of Devices            on      19-JUN-1978 09:45:42.86
Device Device      Device      Err.   Volume           Free  Trans Mount
Name   Status  Characteristics Count   Label          Blocks Count Count
DMA0:  on line  mnt all            1  AARDVARK          2938     1     1
DMA2:  on line  mnt all            0  BACKUPC          20196     2     1
OPA0:  on line                     0
DXA1:  online   mnt for            0  CONSOLE              0     1     1
CRA0:  on line                     0
LPA0:  on line  spl all            0
LPB0:  on line  all                0
TTA0:  on line                     0
TTA1:  on line                     0
TTA2:  on line                     0
  .
  .
  .
TTH7:  off line                    0
XMA0:  on line                     0
DBA1:  on line  MNT                6  SYSTEMUSERS1     17397    14     1
DBA2:  on line  MNT                0  SYSTEMUSERS2     52630    18     1
```

This command displays, for each device in the system:

- Device name
- Device status (indicates whether the device is on line)
- Device characteristics (indicates whether the device is allocated or 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 DMA0:/FULL

  Device  DMA0:            19-JUN-1978 09:45:44:00

          on line
          Mounted
          Error Logging Enabled
          Allocated

     Error count:                   1     Owner process id       00010020
     Operations completed:         38     Owner process name     FACTOR
     Reference count:               1     Default buffer size         512

     Volume label       AARDVARK          Free blocks                2938
     Owner UIC          [001,001]         Transaction count             1
     Volume protection  FF00              Mount count                   1
     Volume status                        Relative volume no.           0
     ACP process name   DBB2ACP           Cluster size                  2
                                          Max.files allowed          4000
```

The SHOW DEVICES command requests a full listing of the status of the RK06/RK07 device DMA0; the information indicates:

- Date and time of day
- Device status
- Error count
- Number of I/O operations completed
- Reference count
- Process identification of the owner of the device
- Process name of the owner of the device
- Default buffer size

For devices with volumes mounted on them, the command displays:

- Volume label
- User identification of the owner of the volume
- Protection code assigned to the volume[1]
- Volume status (indicates whether it is mounted /SYSTEM, or /GROUP)
- Name of the Ancillary Control Process (ACP)
- Relative volume number
- Default cluster size
- Maximum number of files allowed on the volume

For tape devices, the command also displays the default density, the real volume label, and the record size.

1. The volume protection is shown in the order (left to right) of WORLD, GROUP, OWNER, and SYSTEM. Furthermore, each letter is a hexadecimal representation of the access code, in the order (left to right) of DELETE, EXECUTE, WRITE and READ. Thus, in example 2, the volume protection FF00 represents (S:RWED,O:RWED,G,W)

SHOW LOGICAL

Displays all logical names in one or more logical name tables; or displays the current equivalence name assigned to a specified logical name by the ASSIGN, ALLOCATE, DEFINE, or MOUNT commands.

Format

```
SHOW LOGICAL     [logical-name[:]]


Command Qualifiers        Defaults

/ALL                      /ALL
/GROUP
/PROCESS
/SYSTEM
```

Prompts

None.

Command Parameters

logical-name[:]

Specifies a 1- through 63-alphanumeric character logical name for which the equivalence name is to be displayed. The logical name is translated recursively a maximum of 10 times. For each translation, the process, group, and system logical name tables are searched, in that order, and the equivalence name for the first match found is displayed.

If you do not specify a logical name, the command displays all logical names in one or more tables, based on the presence of the /PROCESS, /GROUP, or /SYSTEM qualifiers. If no qualifiers are present and no logical is specified, the command displays all logical names in all logical name tables.

Command Qualifiers

/ALL

Specifies that all logical names in the specified logical name table(s) be displayed. If none of the qualifiers /PROCESS, /GROUP, or /SYSTEM is specified, all names in all logical name tables are displayed.

/GROUP

Indicates, when a logical-name parameter is present, that only the group logical name table is to be searched.

If you specify /ALL either explicitly or by default, all entries in the group logical name table are displayed.

/PROCESS

Indicates, when a logical-name parameter is specified, that only the process logical name table is to be searched.

If you specify /ALL either explicitly or by default, all entries in the process logical name table are displayed.

/SYSTEM

Indicates, when a logical-name parameter is present, that only the system logical name table is to be searched.

If you specify /ALL either explicitly or by default, all names in the system logical name table are displayed.

Examples

1.
```
$ SHOW LOGICAL/PROCESS

  Contents of process logical name table :

  SYS$INPUT = _TTB1:
  SYS$OUTPUT = _TTB1:
  SYS$ERROR = _TTB1:
  SYS$DISK = _DBA3:
  SYS$COMMAND = _TTAB1:
```

   The SHOW LOGICAL command requests a display of the current process logical names. These are the default logical name assignments made by the command interpreter for an interactive process.

2.
```
$ SHOW LOGICAL INFILE
  INFILE = DMB1:PAYROLL.DAT (group)
```

   The SHOW LOGICAL command requests a display of the current equivalence name for the logical name INFILE. The response indicates that the logical name was found in the group logical name table.

3.
```
$ SHOW LOGICAL/GROUP

  Contents of group logical name table :

  group logical name table is empty
```

   The SHOW command requests a display of all current logical names in the group logical name table. The message displayed indicates that there are no logical names in the group logical name table.

4.
```
$ SHOW LOGICAL/SYSTEM SYS$LIBRARY
  SYS$LIBRARY  =  DBB2:[SYSLIB]  (system)
```

The SHOW LOGICAL command requests the equivalence name of SYS$LIBRARY. The response indicates that the default system libraries are in DBB2:[SYSLIB].

5.
```
$ SHOW LOGICAL/GROUP/SYSTEM SYS$DISK
  SYS$DISK = DBA3:[1,1]   (system)
```

The SHOW LOGICAL command is qualified by both the /GROUP and /SYSTEM qualifiers; the response indicates that the logical name SYS$DISK has an equivalence name in the system logical name table.

SHOW MAGTAPE

Displays the current characteristics and status of a specified magnetic tape device.

Format

```
SHOW MAGTAPE    device-name[:]
```

Command Qualifiers	Defaults
None.	None.

Prompts

Device: device-name[:]

Command Parameters

device-name[:]

Specifies the name of the magnetic tape device for which you want to display the characteristics and status.

Examples

1.
```
$ SHOW MAGTAPE MTA0:
  MTA0: UNKNOWN, DENSITY=800, FORMAT=Normal-11
        Odd Parity
```

The SHOW MAGTAPE command requests a display of the characteristics of the device MTA0. It displays the device type, density, and format (default or normal PDP-11).

It can also display the following characteristics:

```
Position Lost          Write-Locked
End-of-Tape            Even Parity
End-of-File            Odd Parity
Beginning-of-Tape
```

Displays the availability of the local node as a member of the network[1] and the names of all nodes that are currently accessible by the local node. The SHOW NETWORK command also displays the address, line and status of the nodes.

Format

```
SHOW NETWORK
```

Command Qualifiers	Defaults
None.	None.

Prompts

None.

Command Parameters

None.

Description

The SHOW NETWORK command displays the name, address, line, and status of the local node and available remote nodes.

If the network is unavailable, the command displays:

```
Network unavailable
```

The possible states are ON or SHUT for the local node and ON, DEV_LOOP (device loopback), and LOOPBACK (line loopback) for the lines. See the DECnet-VAX System Manager's Guide for more details.

1. DECnet-VAX is available under separate license.

Examples

1. `$ SHOW NETWORK`

```
                    Node     Address  Line     State

                    STAR       160    LOCAL     ON
                    GALAXY     161    XMA0      ON
                    VAX4       144    XMB0      DEV_LOOP
```

The SHOW NETWORK command identifies the local node as STAR. The line XMA0 for the remote node GALAXY is on, but the line XMB0 for the remote node VAX4 is in a device loopback state for testing.

SHOW PRINTER

Displays the default characteristics currently defined for a system printer.

Format

```
SHOW PRINTER  device-name[:]
```

Command Qualifiers	Defaults
None.	None.

Prompts

Device: device-name[:]

Command Parameters

device-name[:]

Specifies the name of the printer for which characteristics are to be displayed.

Examples

1.
```
$ SHOW PRINTER LPA0:
LPA0: LP11, WIDTH=132, PAGE=64, NOCR, FF, LOWERCASE
Device spooled to DBB2:
```

The SHOW PRINTER command requests a display of the characteristics of the printer LPA0.

SHOW PROCESS

Displays information about the current process.

Format

```
SHOW PROCESS

Command Qualifiers                 Defaults

/ACCOUNTING                        None.
/ALL
/PRIVILEGES
/QUOTAS
/SUBPROCESSES
```

Prompts

None.

Command Parameters

None.

Command Qualifiers

/ACCOUNTING

Displays accumulated accounting statistics for the current terminal session.

/ALL

Displays all information available, that is, the default information as well as the information displayed by the /ACCOUNTING, /PRIVILEGES, /QUOTAS, and /SUBPROCESSES qualifiers.

/PRIVILEGES

Displays the user privileges that are currently enabled for the process.

/QUOTAS

Displays the process's current quotas. The values displayed reflect any quota reductions resulting from subprocess creation.

/SUBPROCESSES

Displays the process name(s) of any subprocesses owned by the current process; if a hierarchy of subprocesses exists, the command displays the names in hierarchical order.

Examples

1. `$ SHOW PROCESS`

```
19-JUN-1978 11:59:44.13          _TTF3;       User : MALCOLM
Pid : 00160030   Proc. name : MALCOLM         UIC  : [122,001]
Priority :   4   Default file spec. :    DBA1:[MALCOLM.TESTFILES]

Devices allocated :    TTF3:
```

The default output of the SHOW PROCESS command displays:

- Date and time the SHOW PROCESS command is issued
- Device name of the current SYS$INPUT device
- User name
- Process identification number
- Process name
- User identification code (UIC)
- Base execution priority
- Default device
- Default directory
- Devices allocated to the process and volumes mounted, if any

2. `$ SHOW PROCESS/ACCOUNTING`

```
19-JUN-1978 11:59:44.54          _TTF3:       User : MALCOLM

Accounting information:

  Buffered I/O count :      2191        Peak working set size :      180
  Direct I/O count :         263        Peak virtual size :          196
  Page faults :             3131        Mounted volumes :              0
  Elapsed CPU time :     0 00:00:25.67
  Connect time :         0 01:17:47.76
```

3. `$ SHOW PROCESS/PRIVILEGES`

```
19-JUN-1978 11:59:44.71          _TTF3:       User : MALCOLM

Process privileges:

  GRPNAM      may insert in group logical name table
  GROUP       may affect other processes in same group
  PRMCEB      may create permanent common event clusters
  PRMMBX      may create permanent mailbox
  TMPMBX      may create temporary mailbox
```

4. `$ SHOW PROCESS/QUOTAS`

```
19-JUN-1978 11:59:44.86          _TTF3:       User : MALCOLM

Process Quota:

  Account name: DOCUMENT
  CPU limit :                   0 00:00:00.00   Direct I/O limit :     6
  Buffered I/O byte count quota :       12480   buffered I/O limit:    6
  Timer queue entry quota :                10   Open file quota :     16
  Paging file quota :                 2560000   Subprocess quota :     8
  Default page fault cluster :            127   AST limit :           16
```

5.
```
$ SHOW PROCESS/SUBPROCESSES

19-JUN-1978 11:59:45.41        _TTF3:      User : MALCOLM

Subprocesses owned:

- ORION
- CYGNUS
  - LYRA
```

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

Command Qualifiers	Defaults
None.	None.

Prompts

None.

Command Parameters

None.

Examples

1.
```
$ 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 requests a display of the current protection defaults; 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 the current status of entries in the printer and/or batch job queues.

Format

```
SHOW QUEUE [queue-name[:]]
```

Command Qualifiers	Defaults
/ALL	None.
/BATCH	
/BRIEF	
/DEVICE	
/FULL	

Prompts

Queue: queue-name[:]

Command Parameters

queue-name[:]

Specifies the name of a queue you want to display. The queue-name parameter is required if you do not specify either /BATCH or /DEVICE.

Command Qualifiers

/ALL

Displays the names of all jobs in the specified queue. By default, the SHOW QUEUE command displays only current jobs and pending jobs owned by the current process.

/BATCH

Displays entries in all batch job queues.

/BRIEF

Requests a brief listing of information about jobs in the queue. When you specify /BRIEF, only the user name, job number and queue name are displayed.

/DEVICE

Displays the status of jobs in all device queues.

/FULL

Displays the file specifications of each file in each pending job in the queue.

Examples

1. `$ SHOW QUEUE/DEVICE`

```
* Generic Device queue "SYS$PRINT" Flag

Holding Job  261 MALCOLM      BETA    , Pri=4, 19-JAN-1980 12:56

* Device Queue "LPA0" Forms=0, Genprt Lower Flag

Current Job  260 CRAMER       ALPHA   , Pri=4, 19-JAN-1980 12:55
Pending Job  261 HIGGINS      TEMPO   , Pri=4, 19-JAN-1980 12:59
Pending Job  262 HIGGINS      TEMPB   , Pri=4, 19-JAN-1980 13:05

* Device Queue LPB0 Forms=0, Genprt Flag
```

The SHOW QUEUE command displays the status of the printer queues. The first queue, named SYS$PRINT, consists of jobs that are being held. The printer queue, LPA0, is currently processing a job for the user CRAMER; two jobs are pending for the user HIGGINS (who issued the command). There are no jobs in the queue LPB0.

2. `$ SHOW QUEUE SYS$BATCH/FULL`

```
* Batch queue "SYS$BATCH" Joblim=6, Inipri=3, Swap

   WS default: None                    WS quota: None
 Def CPU time: None                Max CPU time: None

  Current Job  263 MALCOLM      SLEEP    Pri=4, 19-JAN-1980 13:08
    WS default: None       WS quota: None      CPU Time: None
  Current Job  261 HIGGINSB     WAITF    Pri=4, 19-JAN-1980 13:02
    WS default: None       WS quota: None      CPU time: None
  Current Job  260 CASEY        RECORD   Pri=4, 19-JAN-1980 12:59
    WS default: None       WS quota: None      CPU Time: None
  Current Job  259 MALCOLM      BATCH1   Pri=4, 19-JAN-1980 12:58
    WS default: None       WS quota: None      CPU time: None
  Holding Job  262 HIGGINS      PROCEDUR Pri=4,  19-JAN-1980 13:11
      Job_count=1, Form_type=0, Characteristics=   3
      DBA1:PROCEDURE.COM;26
  Pending Job  265 HIGGINS      BATCHAVE Pri=4,  19-JAN-1980 13:12
      [1 Intervening Jobs]
  Pending Job  267 HIGGINS      ATTN     Pri=4,  19-JAN-1980 16:59
      DBA1:T.COM;1  Delete
```

The SHOW QUEUE command requests a display of all jobs in the batch job queue. The /FULL qualifier requests the file specifications of pending files in the job. The response indicates a held job and two pending jobs for the user HIGGINS. The job ATTN, consisting of the file T.COM, is marked for deletion after processing.

SHOW QUOTA

Displays the current disk quota that is authorized to and used by a specific user on a specific disk. This display also includes a calculation of the amount of space available and the amount of overdraft that is permitted.

Format

```
SHOW QUOTA

Command Qualifiers          Defaults

/DISK[=device-name[:]]      /DISK=SYS$DISK
/USER=uic
```

Prompts

None.

Command Parameters

None.

Description

This command can identify whether a quota exists for any specific user on a specific disk. Disk quotas are discussed in Chapter 3.

Note that you must have read access to the quota file in order to display the quotas of other users.

The display that results from the SHOW QUOTA command gives the quotas used, authorized, and available in blocks. The amount of overdraft permitted is also shown.

Command Qualifiers

/DISK[=device-name[:]]

Identifies the disk whose quotas are to be examined. By default, SYS$DISK, the current default disk is examined.

/USER=uic

Identifies which user's quotas are to be displayed. The user identification code (UIC) must be specified in square brackets ([ ]), with the group and member numbers separated by commas.

If you omit the UIC, by default your own disk 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 WRKD$
```

   This SHOW QUOTA command 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=DBA1:
  %SYSTEM-F-NODISKQUOTA, no disk quota entry for this UIC
```

   This SHOW QUOTA command displays the fact that the user with UIC [360,007] has no disk quota allocation on device DBA1.

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 WRKD$
```

   This SHOW QUOTA command illustrates a user with an overdrawn quota.

SHOW RMS_DEFAULT

Displays the current default multiblock count and multibuffer count that VAX-11 RMS uses for file operations.

Format

```
SHOW RMS_DEFAULT
```

Command Qualifiers	Defaults
None.	None.

Prompts

None.

Command Parameters

None.

Examples

```
1.  $ SHOW RMS_DEFAULT
                 MULTI-  !              MULTI-BUFFER COUNTS
                 BLOCK   !   Indexed  Relative         Sequential
                 COUNT   !                       Disk   Magtape  Unit Record
    Process       16     !      0        0         0       8        0
    System         4     !      0        0         0       0        0
```

The SHOW RMS_DEFAULT command displays the current process and system default multiblock and multibuffer counts for all types of file.

Displays the status of the image currently executing in the process, if any. The SHOW STATUS command does not affect the image; you can continue the execution of the image after displaying its status.

Format

SHOW STATUS

Command Qualifiers	Defaults
None.	None.

Prompts

None.

Examples

1.
```
$ RUN MYPROG
     .
     .
     .
^Y
$ SHOW STATUS
  Status on  19-JUN-1978 13:16:32.20   Elapsed CPU :    0 00:00:18.37
  Buff. I/O : 1544    Cur. ws. :     180     Open files :          2
  Dir. I/O :   143    Phys. Mem. :    67     Page Faults :       2851
```

The RUN command executes the image MYPROG.EXE. While the program is running, CTRL/Y interrupts it, and the SHOW STATUS command displays its current status.

- Current time and date
- Elapsed CPU time used by the current process
- Number of page faults
- Open file count
- Buffered I/O count
- Direct I/O count
- Current working set size
- Current amount of physical memory occupied

SHOW SYMBOL

Displays the current value of a local or global symbol. Symbols are defined with assignment statements (= command), by passing parameters to a command procedure file, or by the INQUIRE or READ commands.

Format

```
SHOW SYMBOL  [symbol-name]


Command Qualifiers        Defaults

/ALL                      None.
/GLOBAL
/LOCAL
```

Prompts

Symbol: symbol-name

Command Parameters

symbol-name

Specifies the 1- through 255-alphanumeric character name of the symbol whose value you want to display. The symbol-name must begin with an alphabetic character. The SHOW SYMBOL command searches the local symbol table for the current command level, then local symbol tables for preceding command levels, then the global symbol table, for the specified symbol and displays the first match it finds.

If you specify /ALL, you cannot specify a symbol-name.

Command Qualifiers

/ALL

Requests that the current values of all symbols in the specified symbol table be displayed. 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

Requests that only the global symbol table be searched for the specified symbol name.

If you specify /ALL, all names in the global symbol table are displayed.

/LOCAL

Requests that only the local symbol table for the current command level be searched for the specified symbol name.

If you specify /ALL, all names in the local symbol table for the current command level are displayed.

Examples

1.
```
$ SHOW SYMBOL PRINT
  PRINT = PRINT/HOLD
```

   The SHOW SYMBOL command requests that the current value of the symbol name PRINT be displayed. The command interpreter first searches the local symbol table for the current command level, then local symbol tables for preceding command levels, then the global symbol table.

2.
```
$ SHOW SYMBOL/GLOBAL/ALL
  TIM = SHOW TIME
  LOG = @LOG
  $STATUS = %X00000001
  $SEVERITY = 1
```

   The SHOW SYMBOL command requests a display of all symbols defined in the global symbol table. Note that the symbols $STATUS and $SEVERITY, which are maintained by the system, are also displayed.

3.
```
$ SHOW SYMBOL/LOCAL TIM
  TIM =    (undefined)
```

   The SHOW SYMBOL command requests that only the local symbol table be searched for the symbol named TIM. The response indicates that TIM currently has no value.

SHOW SYSTEM

Displays a list of processes in the system and information about the status of each.

Format

SHOW SYSTEM

Command Qualifiers	Defaults
/BATCH	/PROCESS
/NETWORK	/PROCESS
/PROCESS	/PROCESS
/SUBPROCESS	/PROCESS

Prompts

None.

Command Qualifiers

/BATCH

Requests the display of the batch jobs in the system.

By default, all processes are displayed.

/NETWORK

Requests the display of the network processes in the system.

By default, all processes are displayed.

/PROCESS

Displays all processes in the system. This is the default.

/SUBPROCESS

Requests the display of the subprocesses in the system.

By default, all processes are displayed.

Examples

1.
```
$ SHOW SYSTEM
    VAX/VMS  Processes on      19-JAN-1980 15:53:00.71        Uptime 12  04:44:52
    Pid  Process Name      UIC  State Pri Dir. I/O    CPU        Page flts Ph.Mem
    00010000 NULL        000,000 COM   0          0  02:54:03.07         0    0
    00010001 SWAPPER     000,000 HIB  16          0  00:03:25.47         0    0
    00030017 CRAMER      150,020 LEF   4        265  00:00:11.32      1275   58
    00050019 ORION       124,001 COM   4         51  00:00:06.86       721  144  S
    0006001B DEBUG       274,010 LEFO  4         --   swapped out        --   57
    00050023 _JOB350     262,020 COM   4       6272  00:01:08.82      6255  150  B
    0001003C OPERATOR    001,004 LEF  10         62  00:00:00.98        26   35
```

The response displays:

- Process identification
- Process name
- User identification code
- Process state
- Current priority
- Direct I/O count[1]
- Elapsed CPU time[1]
- Number of page faults[1]
- Physical memory occupied[1]
- Process indicator[2]

---

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 message:

    -- swapped out --

2. The letter B indicates a batch job; the letter S indicates a subprocess; the letter N indicates a network process

SHOW TERMINAL

Displays the current characteristics of a specific terminal. Each of these characteristics can be changed with a corresponding option of the SET TERMINAL command.

Format

```
SHOW TERMINAL  [device-name[:]]

Command Qualifiers                 Defaults

/PERMANENT                         None.
```

Prompts

None.

Command Parameters

device-name[:]

Specifies the name of a terminal for which you want the characteristics displayed. If you do not specify a device name, the characteristics of the current device assigned to the logical name SYS$COMMAND are displayed.

Command Qualifiers

/PERMANENT

Displays the current permanent characteristics of the specified terminal.

You must have the LOG_IO or PHY_IO user privilege to use the /PERMANENT qualifier.

Examples

1.
```
$ SHOW TERMINAL

  TTF3: /VT52, WIDTH=80, PAGE=24, OWNER=SELF
        SPEED=(2400,2400), CRFILL=0, LFFILL=0, NO PARITY
        INTERACTIVE, ECHO, TYPEAHEAD, NOESCAPE, NOHOSTSYNC, TTSYNC,
        LOWERCASE, TAB, WRAP, SCOPE, LOCAL, NOHOLDSCREEN,
        NOEIGHTBIT, BROADCAST, NOREADSYNC, NOFORM, HALFDUP,
```

The SHOW TERMINAL command displays the characteristics of the current terminal.

SHOW TRANSLATION

Searches the process, group, and system logical name tables, in that order, for a specified logical name and returns the equivalence name of the first match found.

Format

```
SHOW TRANSLATION  logical-name
```

Command Qualifiers	Defaults
None.	None.

Prompts

```
Log_Name:    logical-name
```

Command Parameters

logical-name

Specifies a 1- through 63-alphanumeric character logical name for which you want to display the translation. The translation is not recursive.

Examples

1.
```
$ SHOW TRANSLATION PAYROLL
  PAYROLL = DMA1:[ACCOUNTS.WORKING]FACTOR1.DAT;37 (process)
```

The SHOW TRANSLATION command displays the current equivalence name of the logical name PAYROLL.

2.
```
$ ASSIGN DBA1:  DISK
$ ASSIGN/GROUP DBB3:  DISK
$ SHOW TRANSLATION DISK
  DISK = DBA1:  (process)
```

The ASSIGN commands place entries for the logical name DISK in both the process and group logical name tables. The SHOW TRANSLATION command shows the logical name for the first entry it finds: the equivalence name placed in the process logical name table.

3.
```
$ RUN ORION
^Y
$ SHOW TRANSLATION TERMINAL
  TERMINAL = _TTF3:  (process)
$ CONTINUE
```

The RUN command 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.

SHOW WORKING_SET

Displays the working set quota and limit assigned to the current process.

Format

```
SHOW WORKING_SET

Command Qualifiers                      Defaults

None.                                   None.
```

Prompts

None.

Command Parameters

None.

Examples

1.
```
$ SHOW WORKING_SET
  Working Set  /Limit=100  /Quota=200  Authorized Quota=200
```

   The response to this command indicates that the current process has a working set limit of 100 pages, a quota of 200 pages, and that the current quota is equal to the authorized limit (200 pages).

Invokes the VAX-11 SORT utility 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 complete details on the qualifiers discussed below and additional information on how to define and control SORT operations, see the VAX-11 SORT User's Guide.

If you use the /RSX11 qualifier, the SORT command invokes the PDP-11 SORT utility program. The SORT/RSX11 command is described under its own heading.

Format

```
SORT   input-file-spec   output-file-spec
```

Command Qualifiers	Defaults
/KEY=(field[,...])	
/PROCESS=type	/PROCESS=RECORD
/SPECIFICATION[=file-spec]	(see text)
/WORK_FILES=n	/WORK_FILES=2

Input File Qualifiers	Defaults
/FORMAT=file-attributes	None.

Output File Qualifiers	Defaults
/ALLOCATION=n	None.
/BUCKET_SIZE=n	
/CONTIGUOUS	
/FORMAT=record-format	
/INDEXED_SEQUENTIAL	
/OVERLAY	
/RELATIVE	
/SEQUENTIAL	

Prompts

File: input-file-spec

Output: output-file-spec

Command Parameters

input-file-spec

Specifies the name of the file whose records are to be sorted. If the file specification does not include a file type, SORT assumes the default file type of DAT.

No wild card characters are allowed in the file specification.

output-file-spec

Specifies the name of the file into which the sorted records are to be written. If an address sort or index sort is selected, output-file specifies the name of the address file.

If the file specification does not include a file type, SORT uses the file type of the input file.

No wild card characters are allowed in the file specification.

Command Qualifiers

/KEY=(field[,...])

Defines a sort key. This qualifier can be specified up to 10 times to define 10 different key fields on which to sort; it must be specified at least once unless /SPECIFICATION is specified.

The field-definition consists of two required keywords that define the position and size of the key field within the record and of several optional keywords that define the type of data within the key field.

The keywords specified must be separated with commas and enclosed in parentheses.

Required Keywords

POSITION:n	Specifies the position of the key within each record, where the first character of each record is position 1.
SIZE:n	Specifies the length of the sort key in characters, bytes, or digits, depending on the key field data type. The valid sizes, based on data types, are:

Data Type	Values for n
character	1 - 255
binary	1, 2, 4
any decimal data type	1 - 31

The total of all key field sizes must be less than 255 bytes.

Optional Keywords

NUMBER:n	Specifies the precedence of the sort key being defined, where 1 represents the primary sort key, 2 represents the secondary sort key, and so on. If this keyword is not specified on the first /KEY qualifier, NUMBER:1 is assumed; if not specified on any subsequent /KEY qualifiers, the default value is the number assigned to the previous key, plus 1. The legal values are 1 through 10.
[CHARACTER BINARY ZONED DECIMAL PACKED_DECIMAL]	Indicates the type of data appearing in the sort key field. If not specified, Sort assumes that data is CHARACTER.
[LEADING_SIGN TRAILING_SIGN]	Indicates whether the sign of a decimal data type key appears at the beginning or end of the key. If the key data type is specified as DECIMAL and neither of these keywords is specified, the default is TRAILING_SIGN.
[OVERPUNCHED_SIGN SEPARATE_SIGN]	Indicates whether the sign of a decimal data type key is superimposed on the decimal value or is separated from the decimal value. If the key data type is specified as DECIMAL and neither of these keywords is specified, the default is OVERPUNCHED_SIGN.
[ASCENDING DESCENDING]	Indicates whether the key is to be sorted into ascending or descending order. If neither of these keywords is specified, the default is ASCENDING.

/PROCESS=type

Defines the type of sort. You can specify one of the following sort types:

ADDRESS	Requests that SORT produce an address file sorted by record keys. The output file can be read as an index to read the original file in the desired sequence.
INDEX	Requests that SORT produce an address file containing the key field of each data record and a pointer to its location in the input file. The output file can be read to randomly access the data in the original file in the desired sequence.
RECORD	Requests SORT to resequence the entire contents of the input file and create an output file containing the reordered records.
TAG	Requests SORT to sort only the record keys, and then to randomly reaccess the input file to create an output file containing the resequenced records.

The default is /PROCESS=RECORD.

/SPECIFICATION[=file-spec]

Indicates that the command and file qualifiers, including key field definitions, are contained in a specification file. If no file specification is included, SORT reads the specification file from SYS$INPUT.

The format and contents of the specification file are described in detail in the VAX-11 SORT User's Guide.

No wild card characters are allowed in the file specification.

/WORK_FILES=n

Specifies the number of temporary work files to be used during the sort process. You can specify 0, or any value from 2 through 10. A value of 0 indicates that no work files are necessary because the data will fit in physical memory.

Input File Qualifiers

/FORMAT=file-attributes

Specifies attributes of the input file to override the existing data that SORT normally obtains through RMS. You can specify one or both of the following keywords:

RECORD_SIZE:n	Specifies, in bytes, the length of the longest record, overriding the record size defined in the file header or label. Specify a record size for an input file when the input file is not on disk or if the longest record size is known to be inaccurate. The maximum record size that can be specified is 16,383 bytes.
FILE_SIZE:n	Defines the size of the file, in blocks. Specify a file size for a file that is not on disk. If no file size is specified for an input file, SORT uses the default value of 1000 blocks. The maximum file size that can be specified is 4,294,967,295 blocks.

Output File Qualifiers

/ALLOCATION=n

Specifies the number of 512-byte blocks to be allocated for the output file. By default, SORT allocates blocks based on the number of records sorted.

The number of blocks specified can be in the range of 1 through 4,294,967,295.

/BUCKET_SIZE=n

Specifies the number of 512-byte blocks per bucket for the output file. The maximum size you can specify is 32 blocks.

If you do not specify a bucket size, the bucket size of the output file is the same as that of the input file (when the output file has the same organization as the input file) or defaults to a value of 1 (when the output file has a different organization than the input file).

/CONTIGUOUS

Controls whether the allocation of disk space for the output file is to be contiguous. If you specify /CONTIGUOUS, you must also specify /ALLOCATION to define the number of blocks to allocate for the output file.

By default, SORT output is not contiguous.

/FORMAT=record-format

Defines the output file record format. You can specify one or more of the following keyword options:

[FIXED[:n] / VARIABLE[:n] / CONTROLLED[:n]]	Defines the output file record format and length, where n is the length of the longest record in the output file. If n is not specified, it defaults to the length of the longest record in the input file. The maximum record size you can specify is 16,383 bytes (less any control bytes).
SIZE:n	Specifies, when CONTROLLED is specified, the size in bytes of the fixed portion of the controlled record. The maximum size of the fixed length control field you can specify is 255 bytes. If CONTROLLED is specified, and no size is specified, Sort uses the default value of 2 bytes.
BLOCK_SIZE:n	Specifies, when the output file is directed to a magnetic tape, the block size, in bytes. By default, SORT uses the block size of the input file if the input file is a tape file. If the input file is not a tape file, Sort uses, by default, the block size specified when the output tape volume was mounted. You can specify a block size in the range of 18 through 65,535. Note, however, that to ensure compatible interchange with most non-DIGITAL systems, the block size should be less than 2048 bytes.

If you do not specify /FORMAT to define the record format of the output file, SORT assumes a default output format based on the sort process selected, as follows:

- If the RECORD or TAG type of sort is selected, the output format and record length default to the format and record length of the input file.
- If the ADDRESS or INDEX type of sort is selected, the output record format defaults to FIXED.

/INDEXED_SEQUENTIAL

Specifies that the output file is in indexed sequential organization. The output file must already exist and be empty. If you specify /INDEXED_SEQUENTIAL, you must also specify /OVERLAY to overlay the existing file.

By default, a record or tag sort creates an output file that is the same organization as the input file. Specify /INDEXED_SEQUENTIAL to create an indexed sequential output file from a sequential or relative input file.

/OVERLAY

Indicates that the existing file is to be overlaid with the sorted records of the input file.

By default, SORT creates a new output file and does not overlay an existing file.

/RELATIVE

Specifies that the output file is in relative file organization.

By default, a record or tag sort results in an output file that has the same organization as the input file. Use /RELATIVE to create a relative output file from a sequential or indexed sequential input file.

/SEQUENTIAL

Specifies that the format of the output file is sequential organization. This is the default for an address or index sort process; for a record or tag sort process, the output file format defaults to the organization of the input file.

Specify /SEQUENTIAL to create an output file that is in a different format from the input file.

Examples

1.
```
$ SORT/KEY=(POSITION:1,SIZE:80) -
$_BOATS.LST  BOATS.TMP
```

   This SORT command sorts the records in the file BOATS.LST and creates an output file named BOATS.TMP. All the records in the input file are sorted in alphanumeric order based on the first 80 characters in each record.

2.
```
$ SORT/KEY=(POSITION:47,SIZE:2)-
$_/KEY=(POSITION:51,SIZE:7)-
$_BOATS.LST  BOATS.BEM
```

   This SORT command specifies two key fields: the first key is in columns 47 and 48 of each input record and the second key is in columns 51 through 57. The output file is named BOATS.BEM.

SORT/RSX11

Invokes the PDP-11 SORT utility program to reorder the records in a file into a defined sequence and to create a new file of the reordered records.

For complete details on the qualifiers discussed below and additional information on how to define and control sort operations, see the PDP-11 SORT Reference Manual.

The /RSX11 qualifier is required.

Format

```
    SORT/RSX11    input-file-spec    output-file-spec


        Additional
    Command Qualifiers              Defaults

    /DEVICE=device-name[:]
    /KEY=(field[,...])              /KEY=(CN1.length)
    /PROCESS=type                   /PROCESS=RECORD
    /SPECIFICATION=file-spec
    /WORK_FILES=n                   /WORK_FILES=5


    File Qualifiers                 Defaults

    /ALLOCATION=ni                  None.
    /BLOCK_SIZE=n
    /BUCKET_SIZE=n
    /CONTIGUOUS
    /FORMAT=(format,size)
    /INDEXED=keys
    /RELATIVE
    /SEQUENTIAL
```

Prompts

File: input-file-spec

Output: output-file-spec

Command Parameters

input-file-spec

Specifies the name of the file whose records are to be sorted. This file must be qualified with either the /FORMAT qualifier or the /INDEXED qualifier to indicate the precise format of the file.

No wild card characters are allowed in the file specification.

output-file-spec

Specifies the name of the file into which the sorted records are to be written.

You can optionally qualify the output-file-spec parameter with the /FORMAT qualifier to indicate the desired output format.

No wild card characters are allowed in the file specification.

Additional Command Qualifiers

/DEVICE=device-name[:]

Specifies the name of the device to be used for work files during sort execution, overriding the device specified when SORT was installed.

/KEY=(field[,...])

Defines the fields in each input record that are the basis for the sort. This qualifier is required.

You can specify up to 10 key fields. Each key field must be specified in the format:

```
[a][b]m.n
```

a defines the way that the data is handled and interpreted. The valid keywords and their meanings are:

B	2's complement binary
C	Alphanumeric
D	If alphabetic, numeric with superimposed sign If FORTRAN numeric, convert to binary
F	2- or 4-word floating point
I	As D above, but with leading + or - sign
J	As D above, but with trailing + or - sign
K	As D above, but with sign overpunched
P	Packed decimal
Z	ASCII zone

If not specified, the default is C, that is, the sort is alphanumeric.

b specifies the general sorting order. You can specify:

N	Ascending order
O	Descending order

If not specified, the default is N, that is, records are sorted in ascending order.

m is a decimal number defining the beginning position of the key field relative to the beginning of each record, with 1 indicating the first position in the record.

This field is required.

n specifies the size of the key field, in bytes.

This field is required.

If you specify more than one key field, separate the specifications with commas and enclose the list in parentheses.

/PROCESS=type

Defines the type of sort. You can specify one of the following options:

ADDRESS	Requests that SORT produce an address file without reordering the input file.
INDEX	Requests that SORT produce an index file containing the key field of each data record and a pointer to its location in the input file.
RECORD	Requests SORT to sort the entire contents of each record in the input file.
TAG	Requests SORT to sort only on the record keys of each record in the input file.

By default, the SORT/RSX11 command produces a record sort.

/SPECIFICATION=file-spec

Specifies the name of a file containing SORT-11 specifications to control the sorting process.

For details on the contents of this file, see the PDP-11 SORT Reference Manual.

No wild card characters are allowed in the file specification.

/WORK_FILES=n

Defines the number of work files to be used during the sorting process, overriding the system-defined default.

You can specify from 3 through 8 work files. The default is 5.

File Qualifiers

/ALLOCATION=n

Specifies the number of 512-byte blocks to allocate for the output file. This qualifier can only be used to qualify the output file.

If no allocation quantity is specified, SORT-11 uses a default allocation quantity based on the type of sorting process.

/BLOCK_SIZE=n

Specifies, when the input and/or output file is a magnetic tape volume, the size of the blocks to be read or written. If not specified, the block size defaults to 512 bytes.

/BUCKET_SIZE=n

Specifies the RMS bucket size allocation for the output file. This qualifier can only be used to qualify the output file parameter.

If no bucket size is specified, SORT-11 uses the bucket size of the input file if the input and output file organizations are similar. If the input file organization is different than the organization requested for the output file, the default is 1.

/CONTIGUOUS

Requests the output file to be written into contiguously allocated disk space. This qualifier can only be used to qualify the output file parameter.

By default, SORT-11 does not create contiguous output files.

/FORMAT=(format,size)

Defines the format and record size of input and output files, where format is one of the keywords listed below and size is the length, in bytes, of the largest record in the file. The valid record formats are:

FIXED
VARIABLE
STREAM
UNKNOWN

This qualifier is required on the input file specification; if not specified for the output file, the output file format defaults to the format of the input file (for RECORD and TAG sort processing). For ADDRESS_ROUTING sort processing, the output file record size is 6 bytes; for INDEX sort processing, the output file record size is 6 bytes plus the size of the input record.

/INDEXED=keys

Specifies that the associated input file is an indexed sequential file and indicates the number of keys in each record in the file.

/RELATIVE

Requests the output file to be in relative file organization. By default, the output file has the same format as the input file.

/SEQUENTIAL

Requests the output file to be in sequential file organization. By default, the output file has the same format as the input file.

Examples

1.
   ```
   $ SORT/RSX11  CUSTOMER.FIL/FORMAT=(FIXED,80) -
   $_ALPHA.SRT/KEY=(1.20)
   SRT -- M:ELAPSED REAL TIME: 00:00:17
   SRT -- M:TOTAL RECORDS SORTED:      1684
   ```

   The SORT command requests a default alphanumeric sort on the records in the file CUSTOMER.FIL. The SORT program sorts the records based on the contents of the first 20 characters in each record and writes the sorted list into the output file ALPHA.SRT.

2.
   ```
   $ SORT/RSX11  CUSTOMER.FIL/FORMAT=(FIXED,80) -
   $_TENURE.FIL/KEY=(29.2,26.2,23.2)
   SRT -- M:ELAPSED REAL TIME: 00:00:25
   SRT -- M:TOTAL RECORDS SORTED:      3245
   ```

   The key fields specified for this SORT command request that the records be sorted first on the 2 characters beginning in column 29, then on the 2 characters beginning in column 26, then on the 2 characters beginning in column 23. If columns 23 through 30 of each record contain a date in the format:

   ```
   dd-mm-yy
   ```

   This command creates an output file with records sorted in ascending order of date.

Terminates execution of:

- A command, image, or command procedure that was interrupted by CTRL/Y
- A command procedure
- A subprocess or a detached process

Format

```
STOP    [process-name]

Command Qualifiers                      Defaults

/IDENTIFICATION=process-id              None.
```

Prompts

None.

Command Parameters

process-name

Specifies the 1- through 15-alphanumeric character-string name of the process to be deleted. The specified process must have the same group number in its user identification code (UIC) as the current process.

If you specify the /IDENTIFICATION qualifier, the process name is ignored. If you specify neither the process-name parameter nor the /IDENTIFICATION qualifier, the image executing in the current process is terminated.

Description

The STOP command causes an abnormal termination of the image currently executing; if the image has declared any exit handling routines, they are not given control. The EXIT command should be used to terminate the image and give exit handler routines control.

Note that when an image is interrupted by CTRL/Y, and the RUN command is issued to execute another image, the interrupted image is also terminated. However, in this case exit handling routines are allowed to execute before the next image is run.

If you interrupt a command procedure by CTRL/Y and you issue 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.

If you specify a process name or process identification, the STOP command terminates the image currently executing in the specified process and deletes the process. If the process is a batch job process, no notification of deletion occurs; the log file for the batch job does not print.

The user privilege GROUP is required to stop other processes in the same group. The user privilege WORLD is required to stop any process in the system.

For more information on how to use commands like this one in command procedures, consult the VAX/VMS Guide to Using Command Procedures.

Command Qualifiers

/IDENTIFICATION=process-id

Specifies the process identification the system assigned to the process when the process was created. When you create a process with the RUN command, the RUN command displays the process identification number of the process it creates.

When you specify the process identification, you can omit leading zeros.

Examples

1.
```
$ RUN  MYPROG
^Y
$ STOP
```

The RUN command begins executing the image MYPROG. Subsequently, CTRL/Y interrupts the execution and the STOP command terminates the image.

2.
```
$ @TESTALL
^Y
$ STOP
```

The @ (Execute Procedure) command executes the procedure TESTALL.COM. CTRL/Y interrupts the procedure and 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 creates a subprocess named LIBRA to execute the image LIBRA.EXE. Subsequently, the STOP command forces 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 as a result of a command or program execution. The STOP command stops all command levels; if this ON command is executed in a command procedure that is executed from within another procedure, control does not return to the outer procedure, but to command level 0.

STOP/ABORT

The STOP/ABORT command aborts a job that is currently being printed. The /ABORT qualifier is required.

Format

```
STOP/ABORT   printer-name[:]


    Additional
Command Qualifiers                              Defaults

None.                                           None.
```

Prompts

Device: printer-name[:]

Command Parameters

printer-name[:]

Specifies the name of the printer queue in which the job was entered.

Description

Use this command only to abort the printing of jobs entered in the system output queues, that is, the line printer or terminal queues.

When you issue the STOP/ABORT command, the job currently being printed is terminated, and the next job in the queue is dequeued, provided you have sufficient privileges to do so. You can always abort your own job, and you can even abort jobs of other users in your group if you have the GROUP user privilege. Otherwise, you need the WORLD or OPER user privileges to abort a job that is not your own.

For more information on how to use commands like this one in command procedures, consult the VAX/VMS Guide to Using Command Procedures.

Examples

1. `$ STOP/ABORT LPA1:`

   This command aborts the job currently printing on line printer LPA1.

STOP/ENTRY

Deletes an entry from a batch queue while it is running. The /ENTRY qualifier is required.

Format

```
STOP/ENTRY=job-number  queue-name[:]
```

Additional Command Qualifiers	Defaults
None.	None.

Prompts

Queue: queue-name[:]

Command Parameters

job-number

Specifies the job number of the job to be deleted from the batch queue.

queue-name[:]

Specifies the name of the queue in which the job was entered.

Description

Use this command to terminate the execution of a batch job while it is running. This command cannot, however, delete a job while it is waiting to be executed. If you want to delete an entry from a device or batch job queue while the entry is waiting to be executed, use the DELETE/ENTRY command.

Note that you can always stop your own job while it is running. You can even stop execution of jobs of other users in your group if you have the GROUP user privilege. Otherwise, you need the WORLD or OPER user privileges to stop a running job that is not your own.

Examples

1.
```
$ STOP/ENTRY=230 SYS$BATCH
```

   The STOP/ENTRY command deletes the job associated with the entry number 230 in the batch queue SYS$BATCH.

STOP/REQUEUE

Stops the printing of the job currently being printed and places that job back at the end of the output queue. The /REQUEUE qualifier is required.

Format

```
STOP/REQUEUE queue-name[:]


    Additional
Command Qualifiers                     Defaults

None.                                  None.
```

Prompts

Queue: queue-name[:]

Command Parameters

queue-name[:]

Specifies the name of the queue to be stopped.

Description

When you requeue a job, that job is placed at the end of the queue with its priority level lowered to 1. The next job in the queue is immediately dequeued for printing.

This command is useful when the line printer runs out of paper while it is printing a job; or when a large job of low priority is currently printing and one or more other jobs in the queue must be printed immediately.

Note that you can always requeue your own job. You can even requeue jobs belonging to other users in your own group if you have the GROUP user privilege. Otherwise, you need the WORLD or OPER user privileges to requeue a job that is not your own.

For more information on how to use commands like this one in command procedures, consult the VAX/VMS Guide to Using Command Procedures.

Examples

1. `$ STOP/REQUEUE LPB0:`

   This command suspends the current print operation on LPB0. It requeues the currently printing job with a priority of 1 at the end of the queue. The printing operation resumes by starting the next job in the queue.

SUBMIT

Enters one or more command procedures in the batch job queue.

Format

```
SUBMIT    file-spec[,...]

Command Qualifiers                Defaults

/AFTER=absolute-time
/CPUTIME=n
/[NO]HOLD                         /NOHOLD
/[NO]IDENTIFY                     /IDENTIFY
/NAME=job-name
/PARAMETERS=(parameter[,...])
/PRIORITY=n
/QUEUE=queue-name[:]
/REMOTE
/WSDEFAULT=n
/WSQUOTA=n

File Qualifiers                   Defaults

/[NO]DELETE                       /NODELETE
```

Prompts

File: file-spec[,...]

Command Parameters

file-spec[,...]

Specifies one or more command procedures to be submitted for batch job execution. You must specify a file name; if you do not specify a file type, the SUBMIT command uses the default file type of COM. If you specify more than one file, you can separate them either with commas (,) or plus signs (+); in either case, the files are concatenated and processed as a single input stream.

If the file specification contains a network node name, the /REMOTE qualifier must be specified.

Full wild card characters are allowed in the file specification. See Section 2.1.6.

Description

A file or files queued by the SUBMIT command are considered a job. The system assigns a unique job number to each job in the system. When you submit a batch job, the system displays both the job number it assigned to the job and the name of the batch job queue in which it entered your job. (If you would like to suppress this display, you can equate SYS$PRINT to NL:, the null device.)

Batch Job Output: When you submit command procedures for processing by the SUBMIT command, all output from the command procedure is written to a file called name.LOG where name is the file name of the first command procedure file in the job. (Use the /NAME qualifier to give the job a different name.) This file is initially written on your default disk; when the batch job completes, the system queues the file to SYS$PRINT and deletes the file after it has printed. If you do not want the log file printed, you can modify the submitted command procedure to assign a nonexistant queue name to the logical name SYS$PRINT.

If multiple procedures are submitted, the job terminates if any procedure exits with an error or fatal error status.

For a description of creating and submitting batch jobs, see the VAX/VMS Guide to Using Command Procedures.

Command Qualifiers

/AFTER=absolute-time

Requests that the job be held until after a specific time. If the specified time has already passed, the job is queued for immediate processing.

Specify the time value according to the rules for entering absolute times given in Section 5.8.

/CPUTIME=n

Defines a CPU time limit for the batch job. You may specify a delta time (Section 5.8.2), the value 0, or the words NONE or INFINITE for n.

Use this qualifier to override the base queue value established by the system manager or the value authorized in your user authorization file, when you need less CPU time than authorized. 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 value or the limit specified on the queue. However, you cannot request more time than permitted by the base limits or your own user authorization file.

/HOLD
/NOHOLD

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 /RELEASE qualifier of the SET QUEUE/ENTRY command.

/IDENTIFY
/NOIDENTIFY

Controls whether the command interpreter displays the job number assigned to the job and the name of the queue in which the job was entered.

By default, the job number and queue name are displayed whenever a job is successfully queued.

/NAME=job-name

Defines a 1- through 8-alphanumeric character-string name to identify the job. The job name is displayed by the SHOW QUEUE command, and is printed on the flag page of the batch job output log, replacing the file name of the log file.

If you do not specify /NAME, the name string defaults to the file name (truncated to eight characters, if necessary) of the first, or only, file in the job.

/PARAMETERS=(parameter[,...])

Specifies from 1 through 8 optional parameters to be passed to the job. The parameters define values to be equated to the symbols named P1, P2, P3, and so on, in each command procedure in the job. The symbols are local to the specified command procedures.

If you specify more than one parameter, separate them with commas and enclose them in parentheses.

The commas delimit the parameters. To specify a parameter that contains any special characters or delimiters, enclose the parameter in quotation marks (").

The total number of characters enclosed in parentheses to specify the parameters, including the comma (,) and quotation mark (") delimiters, must be less than 95 characters.

/PRIORITY=n

Specifies the priority for the specified job. The priority, n, must be in the range of 0 through 31, where 0 is the lowest priority and 31 is the highest.

By default, jobs are queued at the same priority as your current base priority; the user privilege OPER is required to set a priority value that is higher than the base priority of your current process.

/QUEUE=queue-name[:]

Specifies the name of a specific batch job queue to which the job is to be submitted.

/REMOTE

Indicates that the specified command procedure be executed on a remote node. The file specification must contain the name of the node on which the file resides and at which the procedure is to be executed. See the DECnet-VAX User's Guide.

If you specify /REMOTE, you cannot specify any other qualifiers.

/WSDEFAULT=n

Defines a working set default for the batch job. You may specify a positive integer in the range 1 through 65535, 0, or the word NONE for n.

Use this qualifier to override the base queue value established by the system manager or the value authorized in your user authorization file, provided you want to impose a lower value. Specify 0 or NONE if you want the working set value defaulted to either your user authorization file or the working set default specified on the queue. However, you may not request a higher value than your default.

/WSQUOTA=n

Defines the maximum working set size for the batch job. This is the working set quota. You may specify a positive integer in the range 1 through 65535, 0, or the word NONE for n.

Use this qualifier to override the base queue value established by the system manager or the value authorized in your user authorization file, provided you want to impose a lower value. Specify 0 or NONE if you want the working set quota defaulted to either your user authorization file or the working set quota specified on the queue. However, you may not request a higher value than your default.

File Qualifiers

/DELETE
/NODELETE

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. If you specify the /DELETE qualifier following a file specification, only the associated file is deleted after it is processed.

The protection code on the input file(s) must allow delete access to the default user identification code (UIC) of the user who submitted the job.

Examples

1.
```
$ SUBMIT AVERAGE
  Job 112 entered on queue SYS$BATCH
```

The SUBMIT command enters the procedure AVERAGE.COM in the batch job queue. When the batch job completes, the log file AVERAGE.LOG is queued for printing.

2.
```
$ SUBMIT  BACKUP/PARAMETERS=(TXT,DOC,MEM), -
$_AVERAGE, RUNMASTER
  Job 416 entered on queue SYS$BATCH
```

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=BATCH_24/HOLD TESTALL
  Job 467 entered on queue SYS$BATCH
```

The SUBMIT command enters the procedure TESTALL.COM for processing as a batch job, but in a HOLD status. The job will not be released until the SET QUEUE/ENTRY/RELEASE command is issued. The /NAME qualifier requests that the batch job be identified as BATCH_24.

SYNCHRONIZE

Places the process issuing this command in a wait state until a specified batch job completes execution.

Format

```
SYNCHRONIZE    [job-name]

Command Qualifiers                  Defaults

/ENTRY=job-number                   None.
/QUEUE=queue-name[:]
```

Prompts

None.

Command Parameters

job-name

Specifies the 1- through 8-alphanumeric character-string name of the batch job. The job-name corresponds to the name of the job defined by the /NAME qualifier when the job was submitted or to the default job name assigned by the system.

The job must be associated with your current login user name. If you have two or more jobs with the same name, the synchronization occurs against the last job submitted with that name.

If you specify /ENTRY, the job-name parameter is ignored. You must specify either a job name or a job number.

For more information on how to use commands like this one in command procedures, consult the VAX/VMS Guide to Using Command Procedures.

Description

The SYNCHRONIZE command provides batch job synchronization. If a specified job is not currently in the system, the command completes immediately with an error message.

Do not attempt to synchronize more than one process for the completion of a given process; only one process can wait the completion of another process.

When the SYNCHRONIZE command completes, the completion status is the status of the job specified by name or number.

Command Qualifiers

/ENTRY=job-number

Specifies the system-assigned job number of the batch job. The system displays the job number when it successfully queues a job for execution; the job number of a batch job is also displayed when you issue the SHOW QUEUE command.

/QUEUE=queue-name[:]

Specifies the name of the queue on which the job was entered. If not specified, the command assumes that the job is in the default batch job queue, SYS$BATCH.

The queue-name specified is subject to one level of logical name translation.

Examples

1.
```
$ SUBMIT/NAME=PREP FORMAT/PARAMETERS=(SORT,PURGE)
  Job 219 entered on queue SYS$BATCH
$ SUBMIT PHASER
```

The first SUBMIT command submits the command procedure FORMAT.COM for execution and gives the job the job name PREP. The second SUBMIT command queues the procedure PHASER.COM. The procedure PHASER.COM contains the line:

```
$ SYNCHRONIZE PREP
```

When this line is processed, the system verifies whether the batch job named PREP is currently executing. If it is, the procedure PHASER is forced to wait until PREP completes execution.

2.
```
$ SUBMIT/NAME=TIMER   COMP.COM
  Job 214 entered on queue SYS$BATCH
$ SYNCHRONIZE /ENTRY=214
```

In this case, the interactive terminal that issued the SYNCHRONIZE command will be unable to process additional commands until the job named TIMER (Job 214) completes.

TYPE

Displays the contents of a file or group of files on the current output device.

Format

```
TYPE  file-spec[,...]

Command Qualifiers        Defaults

/OUTPUT=file-spec         /OUTPUT=SYS$OUTPUT
```

Prompts

```
File:   file-spec[,...]
```

Command Parameters

file-spec[,...]

Specifies one or more files to be displayed. If you specify a file name and do not specify a file type, the TYPE command uses the default file type of LIS.

If you specify more than one file, separate the file specifications with either commas (,) or plus signs (+). In either case, the files are displayed in the order listed.

You can specify wild card characters in place of the directory file name, file type, or file version number fields. The TYPE command displays all files that satisfy the file description. See Section 2.1.6.

Description

When the TYPE command displays output, you can control the display in any of the following ways:

- To temporarily halt the output and resume it at the line at which it was interrupted, use CTRL/S followed by CTRL/Q.
- To suppress the display but continue command processing, use CTRL/O. If you press CTRL/O again before the 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 is an exception to normal CTRL/O behavior.
- To stop command execution entirely, press CTRL/Y, then issue the EXIT command or any other DCL command (other than CONTINUE).

Command Qualifiers

/OUTPUT=file-spec

Requests that the output from the TYPE command be written to the specified file, rather than SYS$OUTPUT.

Examples

1.
```
$ TYPE  COMMON.DAT
```

   The TYPE command requests that the file COMMON.DAT be displayed at the terminal.

2.
```
$ TYPE *.DAT
    .
    .
    .
^O
    .
    .
    .
^Y
$ STOP
```

   The TYPE command contains a wild card character in place of the file name. Files with file types of DAT are displayed; when CTRL/O is pressed, output of the current file stops and the TYPE command begins displaying the next file. When CTRL/Y interrupts the command, the STOP command terminates the TYPE command.

UNLOCK

Makes accessible a file that became inaccessible as a result of being improperly closed.

Format

```
UNLOCK    file-spec[,...]

Command Qualifiers          Defaults

/[NO]CONFIRM                /NOCONFIRM
/[NO]LOG                    /NOLOG
```

Prompts

```
File:   file-spec[,...]
```

Command Parameters

file-spec[,...]

Specifies one or more files to be unlocked. If you specify more than one file specification, separate them with either commas (,) or plus signs (+).

Wild card characters are allowed in the file specifications. See Section 2.1.6.

Command Qualifiers

/CONFIRM
/NOCONFIRM

Controls whether the UNLOCK command displays the file specification of each file before unlocking it and requests you to confirm whether or not the file actually should be unlocked. If you specify /CONFIRM, you must respond to a prompt with a Y (YES) or a T (TRUE), followed by a carriage return, before the UNLOCK command unlocks the file. If you enter anything else, such as N or NO, the file is not unlocked.

By default, the UNLOCK command does not display the names of files before it unlocks them.

/LOG
/NOLOG

Controls whether the UNLOCK command displays the file specification of each file that it unlocked.

By default, the UNLOCK command does not display the names of files after it unlocks them.

Examples

1.
```
$ TYPE TST.OUT
%TYPE-E-OPENIN, error opening  DBA1:[MAL]TST.OUT;3 as input
-SYSTEM-W-FILELOCKED, file is deaccess locked
$ UNLOCK TST.OUT
$ TYPE TST.OUT
```

The request to type the output file, TST.OUT, returns an error message that indicates the file is locked; the UNLOCK command unlocks it. The TYPE command is used to verify the contents of the file, which may be incomplete.

WAIT

Places the current process in a wait state until a specified period of time has elapsed. The WAIT command is provided for use in command procedures to delay processing of the procedure or of a set of commands in a procedure for a specific amount of time.

Format

```
WAIT    delta-time

Command Qualifiers        Defaults

None.                     None.
```

Prompts

None.

Command Parameters

delta-time

Specifies the time interval to wait. The time must be specified according to the rules for specifying delta time values given in Section 5.8. Note, however, that the delta time can contain only the hours, minutes, seconds, and hundredths of seconds fields; the days part must be omitted. Also, the delta time must begin with the number of hours and not a colon (:), even if the number of hours is zero.

Note that if you issue the WAIT command interactively, the WAIT command does not prompt; however, a time value is required.

Description

If you enter the WAIT command interactively, your current process is placed in a wait state and you cannot enter any more commands. (You can, however, receive unsolicited messages from other processes.) Press CTRL/C or CTRL/Y to restore normal terminal interaction.

For more information on how to use commands like this one in command procedures, consult the VAX/VMS Guide to Using Command Procedures.

Examples

1.
```
$ LOOP:
$ RUN ALPHA
$ WAIT 00:10
$ GOTO LOOP
```

The command procedure executes the program image ALPHA. After the RUN command executes the program, the WAIT command delays execution of the next command for 10 minutes. Note that 00 is specified for the number of hours to avoid beginning the time specification with the colon. After 10 minutes, the GOTO command executes and the procedure loops to the label LOOP and executes the program again. The procedure loops until interrupted or terminated.

If the procedure is executed interactively, it can be terminated by pressing CTRL/C or CTRL/Y and issuing the STOP command or another DCL command that runs a new image in the process. If the procedure is executed in a batch job, it can be terminated with the DELETE/ENTRY command.

WRITE

Writes a record to a specified output file.

Format

```
   WRITE    logical-name  data[,...]

   Command Qualifiers      Defaults

   /ERROR=label            None.
```

Prompts

Log_Name:	logical-name
Symbol:	data[,...]

Command Parameters

logical-name

Specifies the logical name assigned to the file to which a record is to be written. The OPEN command assigns a logical name to a file and places the logical name in the process logical name table.

data[,...]

Specifies data to be written as a single record to the output file. You can specify one or more symbol names, character strings enclosed in quotation marks, or literal numeric values. The items specified in the data list must be separated by commas; the command interpreter concatenates the items into a single record and writes the record to the output file.

The maximum size of any record that can be written is 255 bytes.

Description

The WRITE command can write records only to sequential files, and cannot be used to append new records at the end of existing files.

Before a file can be written, it must be opened with the OPEN command and assigned a logical name. 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.

For a description of the VAX/VMS file handling commands, see the VAX/VMS Guide to Using Command Procedures.

Command Qualifiers

/ERROR=LABEL

Specifies a label on a line in the command procedure to receive control if the write request results in an error. If no error routine is specified and an error occurs during the writing of the file, the command procedure continues execution at the next line in the file, as it does if no error occurs.

The error routine specified for this qualifier takes precedence over any action statement indicated in an ON command. 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 reserved global symbol $STATUS contains a successful completion status value.

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
$ INQUIRE ID "Assign Test-id Number"
$ WRITE/ERROR=END_LOOP  OUTPUT_FILE  "Test-id is ",ID
$ WRITE/ERROR=END_LOOP  OUTPUT_FILE  ""
$ !
$ WRITE_LOOP:
    .
    .
    .
$ GOTO WRITE_LOOP
$ END_LOOP:
$ !
$ CLOSE OUTPUT_FILE
$ PRINT/DELETE TESTFILE.DAT
```

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 define processing that results in additional data written to the file. The label END_LOOP is also used as the target of the /ERROR qualifier on the WRITE command; the CLOSE and PRINT commands at this label close the output file and queue a copy of the file to the system printer. The output file TESTFILE.DAT is deleted after printing.

APPENDIX A

FOREIGN COMMAND FEATURE OF DCL

The command interpreter allows you to define foreign commands. A foreign command is a command that is not known to the command interpreter but that can be executed by entering a command string.

The command interpreter provides the following mechanisms so you can execute your programs as foreign commands and pass variable data to them at execution time:

- To specify parameters when an image is run, you must use an assignment statement to define a command name to be used instead of a RUN command to execute an image.
- To obtain the parameters, the image must request the parameter string from the command interpreter and must perform all string parsing and analysis itself.

Each of these mechanisms is described in detail below.

A.1 DEFINING A FOREIGN COMMAND

Use the following syntax of an assignment statement to define a foreign command:

```
     $ symbol-name :=[=] $image-file-spec
```

symbol-name
The name by which you want to invoke the image.

$image-file-spec
The file specification of the image to be executed. The image-file-spec must contain a device name and a file name; optionally, you can specify a directory name, a file type, or a file version. In general, the image-file-specification should contain a directory name. The default device and directory name is SYS$SYSTEM, the default file type is EXE, and the default file version number is the highest version.

The dollar sign ($) preceding the image-file-specification is required.

After you have defined a foreign command as shown above, the request to execute the image is implicit in the symbol definition. When this symbol-name is specified as the first token, or item, in a command, the command interpreter executes the specified image. For example:

```
     $ PROCESS := $DB1:[MALCOLM.PROG]CREPROCES
     $ PROCESS
```

In this example, the symbol-name PROCESS is defined as a foreign command. When PROCESS is specified as the first token in a command, you can specify any data following it. For example:

```
     $ PROCESS ORION
```

This command string passes the string ORION to the executing image. The image must obtain the parameter string. The image must also perform any parsing or evaluation of the command string; the command interpreter does not parse the line.

Note that during command input, the command interpreter performs all symbol substitution requested by apostrophes (') in the command string. Thus, if you use symbols preceded by apostrophes to specify parameters in a command string, substitution of these symbols occurs before the resulting command string is passed to the program.

A.2 ABBREVIATING THE FOREIGN COMMAND

You can use abbreviated forms of foreign commands if you define them using the abbreviation punctuation character, the asterisk (*), as described in Section 5.11. For example, to abbreviate the foreign command, DISPLAY, define it as:

```
     $ DISP*LAY := $DISPLAY
```

Then, the DISPLAY utility is executed whenever any of the following versions of the symbolic name is used:

```
     DISP
     DISPL
     DISPLA
     DISPLAY
```

A.3 OBTAINING A PARAMETER STRING FROM THE COMMAND INTERPRETER

To obtain a parameter string passed with a foreign command, you can call the Run-Time Library routine LIB$GET_FOREIGN. The following summary and examples help illustrate how to use LIB$GET_FOREIGN. For more information, see the VAX-11 Run-Time Library Reference Manual.

Format:

```
     status = LIB$GET_FOREIGN (param_string [, prompt [, outlen]])
```

Parameters:

param_string	- The address of a string descriptor to which the parameter string will be returned. The string may be fixed or dynamic.
prompt	- Optional. The address of a string descriptor containing a prompt string. If this parameter is present, and no parameter string is available from the command interpreter, a line is read from SYS$INPUT using the supplied prompt. If the prompt parameter is omitted, a zero-length string is returned to param_string.
outlen	- Optional. The address of a longword integer in which is stored the length of the parameter string or SYS$INPUT line that is obtained. This is useful if param_string is fixed length.

Sample VAX-11 MACRO program:

```
        $DSCDEF                         ; Define descriptor codes

PARAM:                                  ; Descriptor of returned string
        .WORD    0                      ; Initial length is zero
        .BYTE    DSC$K_DTYPE_T          ; Text string
        .BYTE    DSC$K_CLASS_D          ; Dynamic string
        .LONG    0                      ; Address of string (will be filled
                                        ; in by LIB$GET_FOREIGN)

PROMPT:                                 ; Prompt to use if no parameter string
        .ASCID   /Command: /
;+
; Functional description:
;
;       Sample program to demonstrate foreign command lines.  Gets
;       line from command interpreter or SYS$INPUT and displays line
;       on SYS$OUTPUT.
;-
         .ENTRY   GETCMDLIN, ^M<>
         PUSHAQ   PROMPT                ; Adress of prompt descriptor
         PUSHAQ   PARAM                 ; Address of parameter descriptor
         CALLS    #2, G^LIB$GET_FOREIGN
         PUSHAQ   PARAM                 ; Display parameter line on SYS$OUTPUT
         CALLS    #1, G^LIB$PUT_OUTPUT
         RET
         .END      GETCMDLIN
```

Sample VAX-11 FORTRAN program:

```
      PROGRAM GETCMDLIN
      CHARACTER*80 PARAM
      INTEGER*4 LENGTH
      CALL LIB$GET_FOREIGN (PARAM, 'Command: ', LENGTH)
      TYPE *, PARAM(1:LENGTH)
      END
```

INDEX

Please cut along this line.

VAX/VMS
Command Language
User's Guide
AA-D023B-TE

READER'S COMMENTS

NOTE: This form is for document comments only. DIGITAL will use comments submitted on this form at the company's discretion. If you require a written reply and are eligible to receive one under Software Performance Report (SPR) service, submit your comments on an SPR form.

Did you find this manual understandable, usable, and well-organized? Please make suggestions for improvement.

Did you find errors in this manual? If so, specify the error and the page number.

Please indicate the type of reader that you most nearly represent.

- ☐ Assembly language programmer
- ☐ Higher-level language programmer
- ☐ Occasional programmer (experienced)
- ☐ User with little programming experience
- ☐ Student programmer
- ☐ Other (please specify)

Name ____________ Date ____________

Organization ____________

Street ____________

City ____________ State ____________ Zip Code or Country ____________

– – – 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

BSSG PUBLICATIONS TW/A14
DIGITAL EQUIPMENT CORPORATION
1925 ANDOVER STREET
TEWKSBURY, MASSACHUSETTS 01876

– – Do Not Tear - Fold Here –