

# **VAX/VMS User's Manual**

Order Number: AI-Y517A-TE

**April 1986**

This manual is an overview of the general use of the VAX/VMS operating system.

**Operating System and Version:** VAX/VMS Version 4.4

**Software Version:** VAX/VMS Version 4.4

**digital equipment corporation  
maynard, massachusetts**

---

April 1986

The information in this document is subject to change without notice and should not be construed as a commitment by Digital Equipment Corporation. Digital Equipment Corporation assumes no responsibility for any errors that may appear in this document.

The software described in this document is furnished under a license and may be used or copied only in accordance with the terms of such license.

No responsibility is assumed for the use or reliability of software on equipment that is not supplied by Digital Equipment Corporation or its affiliated companies.

Copyright ©1986 by Digital Equipment Corporation

All Rights Reserved.

Printed in U.S.A.

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

The following are trademarks of Digital Equipment Corporation:

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

**digital**

ZK3144

---

**HOW TO ORDER ADDITIONAL DOCUMENTATION**  
**DIRECT MAIL ORDERS**

**USA & PUERTO RICO\***

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

**CANADA**

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

**INTERNATIONAL**

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

In Continental USA and Puerto Rico call 800-258-1710.

In New Hampshire, Alaska, and Hawaii call 603-884-6660.

In Canada call 800-267-6215.

\* Any prepaid order from Puerto Rico must be placed with the local Digital subsidiary (809-754-7575).

Internal orders should be placed through the Software Distribution Center (SDC), Digital Equipment Corporation, Westminister, Massachusetts 01473.

---

This document was prepared using an in-house documentation production system. All page composition and make-up was performed by T<sub>E</sub>X, the typesetting system developed by Donald E. Knuth at Stanford University. T<sub>E</sub>X is a registered trademark of the American Mathematical Society.



# Contents

## Preface

xv

## Chapter 1 Interaction with the System

1.1	Logging in to the System . . . . .	1-1
1.1.1	Automatic Login . . . . .	1-2
1.1.2	Dialing in . . . . .	1-2
1.1.3	Logging in over the Network . . . . .	1-3
1.1.3.1	Access and Login . . . . .	1-5
1.1.3.2	Terminating a Remote Session . . . . .	1-6
1.1.4	Logging out of the System . . . . .	1-6
1.2	Using the Terminal . . . . .	1-7
1.2.1	Entering Text . . . . .	1-7
1.2.2	Controlling Output . . . . .	1-8
1.2.2.1	Suspending/Resuming Terminal Display . . . . .	1-8
1.2.2.2	Formatting Screen Output with Escape Sequences . . . . .	1-8
1.3	Setting the Terminal's Physical Attributes . . . . .	1-10
1.3.1	VT200 Series . . . . .	1-10
1.3.1.1	The Set-Up Directory . . . . .	1-11
1.3.1.2	Changing Attributes with Set-Up . . . . .	1-12
1.3.1.3	Critical Attributes . . . . .	1-14
1.3.2	VT100 Series . . . . .	1-14
1.3.2.1	SET-UP Mode A . . . . .	1-15
1.3.2.2	SET-UP Mode B . . . . .	1-16
1.4	Commands and Utilities . . . . .	1-18
1.4.1	DCL Command Format . . . . .	1-19
1.4.2	Parameters . . . . .	1-20
1.4.3	Qualifiers . . . . .	1-21
1.4.4	Prompts . . . . .	1-22
1.4.5	Interactive Commands . . . . .	1-23
1.4.6	Interrupting Commands . . . . .	1-23
1.4.7	Editing Command Lines . . . . .	1-24
1.4.8	Help . . . . .	1-26

1.5	Shortcuts for Entering Commands . . . . .	1-27
1.5.1	Symbols . . . . .	1-27
1.5.2	Command Procedures . . . . .	1-28
1.5.3	Key Definitions . . . . .	1-29
1.5.3.1	Definable Keys . . . . .	1-30
1.5.3.2	Key States . . . . .	1-30
1.5.3.3	Examining and Deleting Keys . . . . .	1-31
1.5.4	Recalling Command Lines . . . . .	1-32
1.6	User Environment . . . . .	1-33
1.6.1	Programs . . . . .	1-33
1.6.2	Processes . . . . .	1-34
1.6.3	System Implementation . . . . .	1-36
1.6.3.1	Memory . . . . .	1-36
1.6.3.2	Images and Working Sets . . . . .	1-37
1.6.3.3	Processes and the Balance Set . . . . .	1-38
1.6.3.4	Resident System . . . . .	1-39
1.6.3.5	System Parameters . . . . .	1-39
1.7	Subprocesses . . . . .	1-39
1.7.1	Exiting from a Subprocess . . . . .	1-41
1.7.2	Subprocess Context . . . . .	1-42
1.7.3	Subprocess Execution . . . . .	1-43
1.8	Batch Jobs . . . . .	1-43
1.8.1	Submitting a Batch Job . . . . .	1-44
1.8.2	Controlling a Batch Job . . . . .	1-45
1.8.3	Batch Job Output . . . . .	1-46
1.8.4	Restarting Batch Jobs . . . . .	1-46
1.9	Error Conditions . . . . .	1-48
1.9.1	Commands That Do Not Change \$STATUS . . . . .	1-48
1.9.2	System Error Messages . . . . .	1-49
1.10	Using the Mail Utility . . . . .	1-49
1.10.1	Sending Mail . . . . .	1-49
1.10.1.1	Sending a Mail Message . . . . .	1-50
1.10.1.2	Sending a File . . . . .	1-51
1.10.1.3	Setting the Default Editor . . . . .	1-51
1.10.1.4	Sending a Message Between Systems . . . . .	1-52
1.10.1.5	Sending a Message to a Distribution List . . . . .	1-52
1.10.2	Reading Mail . . . . .	1-53
1.10.2.1	New Messages . . . . .	1-53
1.10.2.2	Old Messages . . . . .	1-54
1.10.3	Creating a File from a Mail Message . . . . .	1-55
1.10.4	Deleting Mail . . . . .	1-55

1.10.5	Organizing Mail . . . . .	1-56
1.10.5.1	Creating and Modifying Folders . . . . .	1-57
1.10.5.2	Deleting Folders . . . . .	1-57
1.10.5.3	Creating, Accessing, and Deleting Mail Files . . . . .	1-58
1.10.5.4	Selecting Folders . . . . .	1-58
1.10.6	Using a Mail Subdirectory . . . . .	1-59
1.10.7	Using the Mail Keypad . . . . .	1-59

## Chapter 2 Storage and Output of Data

2.1	Devices . . . . .	2-1
2.1.1	Physical Device Names . . . . .	2-2
2.1.2	Logical Device Names . . . . .	2-2
2.1.3	Generic Device Names . . . . .	2-3
2.2	File Operations on Disks . . . . .	2-3
2.2.1	File Specifications . . . . .	2-3
2.2.1.1	Setting Device and Directory Defaults . . . . .	2-4
2.2.1.2	Using Default File Specifications . . . . .	2-5
2.2.1.3	Matching Wildcards (* and %) . . . . .	2-6
2.2.1.4	Search Wildcards (... and -) . . . . .	2-7
2.2.2	Directory Operations . . . . .	2-8
2.2.2.1	Displaying Directories . . . . .	2-9
2.2.2.2	Creating Directories . . . . .	2-10
2.2.2.3	Deleting Directories . . . . .	2-10
2.2.3	File Operations . . . . .	2-11
2.2.3.1	File Characteristics . . . . .	2-12
2.2.3.2	Creating and Modifying Files . . . . .	2-14
2.2.3.3	Examining Files . . . . .	2-17
2.2.3.4	Deleting Files . . . . .	2-17
2.2.3.5	Common Qualifiers . . . . .	2-18
2.2.4	System Directories and Files . . . . .	2-19
2.3	Logical Names . . . . .	2-20
2.3.1	Logical Name Definition . . . . .	2-20
2.3.1.1	Defining Logical Names . . . . .	2-21
2.3.1.2	Deassigning Logical Names . . . . .	2-22
2.3.1.3	Displaying Logical Names . . . . .	2-22
2.3.2	Logical Name Translation . . . . .	2-23
2.3.2.1	Iterative Translation . . . . .	2-23
2.3.2.2	Noniterative Translation . . . . .	2-24
2.3.2.3	Concealed Translation of Device Names . . . . .	2-24
2.3.2.4	Search List Translation . . . . .	2-24
2.3.3	Scope and Precedence of Logical Names . . . . .	2-25
2.3.3.1	Logical Name Table Directories . . . . .	2-26

2.3.3.2	Precedence . . . . .	2-28
2.3.3.3	Logical Name Table Creation . . . . .	2-29
2.3.3.4	Access Modes . . . . .	2-30
2.3.4	System-Created Logical Names . . . . .	2-30
2.3.4.1	Process-Permanent Logical Names . . . . .	2-30
2.3.4.2	System-Permanent Logical Names . . . . .	2-31
2.4	Printing Files . . . . .	2-32
2.4.1	Controlling a Print Job . . . . .	2-33
2.4.2	Controlling a Print Queue . . . . .	2-34
2.4.2.1	Stopping and Restarting a Queue . . . . .	2-34
2.4.2.2	Creating a Print Queue . . . . .	2-34
2.4.2.3	Specifying Print Features and Restrictions . . . . .	2-35
2.4.2.4	Using the FORM=type Keyword . . . . .	2-35
2.4.3	Using Multiple Print Queues . . . . .	2-36
2.4.3.1	One Printer with Multiple Queues . . . . .	2-36
2.4.3.2	One Queue with Multiple Printers . . . . .	2-37
2.5	Sorting and Merging Files . . . . .	2-37
2.5.1	Record Sorting . . . . .	2-37
2.5.1.1	Single Key . . . . .	2-38
2.5.1.2	Multiple Keys . . . . .	2-39
2.5.2	Other Types of Sorting . . . . .	2-39
2.5.3	Character Data Files . . . . .	2-40
2.5.4	Noncharacter Data Files . . . . .	2-40
2.5.5	Terminal Input . . . . .	2-41
2.5.6	Output File Organization . . . . .	2-41
2.5.7	Batch Job Submission . . . . .	2-41
2.5.8	Specification Files . . . . .	2-42
2.5.8.1	Creating or Modifying a Collating Sequence . . . . .	2-43
2.5.8.2	Reformatting Records in the Output File . . . . .	2-44
2.5.8.3	Conditionally Selecting Key and Data Fields . . . . .	2-44
2.5.9	Merging Files . . . . .	2-45
2.6	Using Libraries . . . . .	2-45
2.6.1	Creating Text Libraries . . . . .	2-46
2.6.2	Creating Help Libraries . . . . .	2-46
2.6.2.1	Creating Help Modules . . . . .	2-47
2.6.2.2	Naming Help Libraries . . . . .	2-49
2.6.3	Displaying Libraries . . . . .	2-49
2.6.4	Deleting Libraries . . . . .	2-50
2.6.5	Adding Library Modules . . . . .	2-50
2.6.6	Deleting Library Modules . . . . .	2-51
2.6.7	Modifying Library Modules . . . . .	2-51
2.7	Transferring Files Between Systems . . . . .	2-52

2.7.1	Reading Files from Another System . . . . .	2-53
2.7.2	Writing Files to Another System . . . . .	2-53
2.7.3	Access Restrictions . . . . .	2-54

## Chapter 3 Editing Files with the EDT Editor

3.1	Invoking and Terminating EDT . . . . .	3-1
3.1.1	Invoking EDT . . . . .	3-2
3.1.2	Terminating EDT . . . . .	3-2
3.1.2.1	Saving Your Edits . . . . .	3-3
3.1.2.2	Discarding Your Edits . . . . .	3-3
3.2	Entering EDT Commands . . . . .	3-3
3.2.1	Entering EDT Line Commands . . . . .	3-3
3.2.2	Entering Keypad Commands . . . . .	3-4
3.2.3	Canceling EDT Commands . . . . .	3-6
3.3	Getting Help in EDT . . . . .	3-6
3.3.1	Getting HELP on Keypad-Editing Commands . . . . .	3-7
3.3.2	Getting HELP on Line-Editing Commands . . . . .	3-7
3.3.3	Getting HELP on Nokeypad-Editing Commands . . . . .	3-7
3.4	Changing Editing Modes . . . . .	3-7
3.4.1	Changing from Keypad to Line Editing . . . . .	3-8
3.4.2	Changing from Line to Keypad Editing . . . . .	3-8
3.4.3	Entering Line-Editing Commands from Keypad Mode . . . . .	3-8
3.5	Recovering from Interruptions . . . . .	3-9
3.6	EDT Keypad Editing . . . . .	3-10
3.6.1	Manipulating the Cursor . . . . .	3-10
3.6.1.1	Moving the Cursor by Small Units . . . . .	3-10
3.6.1.2	Moving the Cursor by Large Units . . . . .	3-13
3.6.1.3	Moving the Cursor to the Beginning or End of the Buffer . . . . .	3-13
3.6.1.4	Changing the Cursor's Direction . . . . .	3-14
3.6.2	Inserting Text . . . . .	3-16
3.6.3	Deleting and Restoring Text . . . . .	3-16
3.6.3.1	Deleting and Restoring Characters . . . . .	3-17
3.6.3.2	Deleting and Restoring Words . . . . .	3-17
3.6.3.3	Deleting and Restoring Lines . . . . .	3-18
3.6.3.4	Deleting and Restoring Large Sections . . . . .	3-19
3.6.4	Locating Text . . . . .	3-20
3.6.5	Substituting Text . . . . .	3-22
3.6.6	Moving Text . . . . .	3-24

3.6.6.1	Moving Text Within the File . . . . .	3-24
3.6.6.2	Moving Text Between Files . . . . .	3-29
3.6.7	Using Multiple Buffers . . . . .	3-29
3.7	Controlling EDT Sessions . . . . .	3-32
3.7.1	Startup Command Files . . . . .	3-32
3.7.2	Controlling Screen Format with SET Commands . . . . .	3-33
3.7.3	Controlling Editing Functions with SET Commands . . . . .	3-34
3.7.4	Defining Keys . . . . .	3-35
3.7.5	Defining EDT Macros . . . . .	3-36

## Chapter 4 Using DIGITAL Standard Runoff

4.1	Formatting Text . . . . .	4-3
4.1.1	Filling and Justifying Text . . . . .	4-4
4.1.2	Adjusting Margins and Centering Text . . . . .	4-6
4.1.3	Writing Paragraphs . . . . .	4-7
4.1.4	Writing Literal Text . . . . .	4-8
4.1.5	Writing Lists . . . . .	4-9
4.1.5.1	Numbered Lists . . . . .	4-10
4.1.5.2	Bulleted Lists . . . . .	4-11
4.1.5.3	Nested Lists . . . . .	4-11
4.1.5.4	Letters and Roman Numerals . . . . .	4-12
4.1.6	Leaving Space on a Page . . . . .	4-12
4.1.7	Writing Notes . . . . .	4-13
4.1.8	Writing Footnotes . . . . .	4-13
4.1.9	Bolding and Underlining Text . . . . .	4-14
4.2	Laying Out a Document . . . . .	4-15
4.2.1	Chapters and Appendixes . . . . .	4-16
4.2.2	Sections . . . . .	4-16
4.2.3	Running Heads . . . . .	4-18
4.2.4	Pagination . . . . .	4-19
4.3	Processing DSR Files . . . . .	4-19
4.3.1	Producing a Table of Contents . . . . .	4-20
4.3.2	Producing an Index . . . . .	4-21
4.3.3	Printing Output Files . . . . .	4-23

## Chapter 5 Data Representation

5.1	Data Storage . . . . .	5-1
5.1.1	Character Data . . . . .	5-3
5.1.2	Numeric Data . . . . .	5-5
5.1.3	Logical Data . . . . .	5-6
5.2	Expressions . . . . .	5-7
5.2.1	Character Expressions . . . . .	5-7
5.2.2	Numeric Expressions . . . . .	5-9
5.2.3	Logical Expressions . . . . .	5-11
5.2.4	Combined Operations and Precedence . . . . .	5-12
5.3	Lexical Functions . . . . .	5-13
5.4	Date and Time . . . . .	5-14
5.4.1	Absolute Time . . . . .	5-14
5.4.2	Delta Time . . . . .	5-15
5.4.3	Absolute and Delta Time Combinations . . . . .	5-15
5.5	Symbols . . . . .	5-16
5.5.1	Creation and Deletion . . . . .	5-17
5.5.2	Symbol Access Control . . . . .	5-18
5.5.3	Symbol Substitution . . . . .	5-18
5.5.4	Substring Substitution . . . . .	5-20
5.5.5	Use of Symbols . . . . .	5-21

## Chapter 6 Command Procedures

6.1	Format . . . . .	6-2
6.2	Execution . . . . .	6-2
6.2.1	Nesting Command Levels . . . . .	6-3
6.2.2	Exiting from Command Procedures . . . . .	6-3
6.3	Passing Data . . . . .	6-4
6.3.1	Using Parameters To Pass Data . . . . .	6-4
6.3.2	The INQUIRE Command . . . . .	6-6
6.3.3	The READ Command . . . . .	6-7
6.3.4	Obtaining Data from SYS\$INPUT . . . . .	6-7
6.4	Returning Data . . . . .	6-8
6.5	Displaying Data . . . . .	6-9
6.5.1	Displaying Literals and Symbols . . . . .	6-9
6.5.2	Displaying Text . . . . .	6-10
6.5.3	Displaying Files . . . . .	6-10



6.6	Inputting and Outputting Data from Files (File I/O) . . . .	6-10
6.6.1	Writing to a File . . . . .	6-10
6.6.2	Reading from a File . . . . .	6-12
6.6.3	Modifying a File . . . . .	6-13
6.6.3.1	Minor Modifications . . . . .	6-13
6.6.3.2	Major Modifications . . . . .	6-15
6.6.4	Handling I/O Errors . . . . .	6-16
6.7	Using Logic to Control the Flow of Execution . . . . .	6-16
6.7.1	Designing Complicated Command Procedures . . . . .	6-16
6.7.2	Coding Complicated Command Procedures . . . . .	6-18
6.7.2.1	Conditional Code . . . . .	6-19
6.7.2.2	Case Statements . . . . .	6-20
6.7.2.3	Loops . . . . .	6-21
6.7.2.4	Subroutines . . . . .	6-23
6.7.3	Testing and Debugging . . . . .	6-25
6.8	Handling Errors and CTRL/Y Interrupts . . . . .	6-28
6.8.1	The ON Command . . . . .	6-28
6.8.2	The SET [NO]ON Command . . . . .	6-29
6.8.3	CTRL/Y Interrupts . . . . .	6-30
6.9	Cleanup Operations . . . . .	6-30

## Chapter 7 Accounts and Security

7.1	User Accounts . . . . .	7-1
7.1.1	User Authorization File . . . . .	7-1
7.2	Protection . . . . .	7-1
7.2.1	UIC-Based Protection . . . . .	7-2
7.2.1.1	User Identification Code . . . . .	7-3
7.2.1.2	Ownership and Access Categories . . . . .	7-4
7.2.1.3	Protection Masks . . . . .	7-5
7.2.1.4	Securing User Data and Devices . . . . .	7-5
7.2.2	ACL-Based Protection . . . . .	7-6
7.2.2.1	Object Types . . . . .	7-6
7.2.2.2	Identifiers . . . . .	7-7
7.2.2.3	Access Control Entries (ACE) . . . . .	7-8
7.2.2.4	IDENTIFIER ACES . . . . .	7-8
7.2.2.5	Rights List . . . . .	7-9
7.2.2.6	DEFAULT_PROTECTION ACES . . . . .	7-10
7.2.2.7	ALARM_JOURNAL ACES . . . . .	7-10
7.2.3	Protection of Files . . . . .	7-11
7.2.3.1	Default File Protection . . . . .	7-11
7.2.3.2	Explicit File Protection . . . . .	7-12



7.2.3.3	Protection of Directories . . . . .	7-12
7.2.3.4	Protection of Mail Files . . . . .	7-12
7.2.4	Protection of Disk Volumes . . . . .	7-13
7.2.5	Protection of Devices . . . . .	7-13
7.2.6	Displays of Ownership and Protection . . . . .	7-14
7.3	Creating and Deleting ACLs . . . . .	7-14
7.3.1	Using the SET ACL Command . . . . .	7-15
7.3.2	ACL Editor . . . . .	7-16
7.3.2.1	Using Prompts . . . . .	7-18
7.3.2.2	Moving the Cursor . . . . .	7-18
7.3.2.3	Entering and Deleting Data . . . . .	7-19
7.3.2.4	Processing an ACE . . . . .	7-20

## **Appendix ACL Access Control Lists**

ACL.1	Format of Access Control Entries . . . . .	ACL-1
ACL.2	EDIT/ACL Command . . . . .	ACL-4
ACL.3	ACL Editor Commands . . . . .	ACL-6
ACL.3.1	Keypad Diagram . . . . .	ACL-6
ACL.3.2	Keypad Commands . . . . .	ACL-7
ACL.3.3	Control Keys . . . . .	ACL-15
ACL.4	DCL Commands That Affect ACLs . . . . .	ACL-15

## **Appendix CHAR Character Sets**

CHAR.1	ASCII Character Set . . . . .	CHAR-1
CHAR.2	ASCII and DEC Multinational Character Set Tables . . . . .	CHAR-2
CHAR.3	DEC Multinational Character Set . . . . .	CHAR-5

## **Appendix DCL DCL Commands**

### **Appendix DSR Digital Standard Runoff**

DSR.1	DCL Commands for Invoking DSR . . . . .	DSR-1
DSR.2	DSR Commands . . . . .	DSR-10
DSR.3	DSR Flags . . . . .	DSR-45
DSR.4	Index Formatting . . . . .	DSR-47

## Appendix EDT EDT Editor

EDT.1	EDIT Command . . . . .	EDT-1
EDT.2	EDT Keypad Editing . . . . .	EDT-3
EDT.2.1	Keypad Commands . . . . .	EDT-3
EDT.2.2	Keyboard Keys . . . . .	EDT-15
EDT.2.3	VT200 Supplemental Editing Keypad . . . . .	EDT-16
EDT.2.4	Control Keys . . . . .	EDT-17
EDT.3	Line Editing . . . . .	EDT-21
EDT.3.1	Range Specifications . . . . .	EDT-21
EDT.3.2	Line-Editing Commands . . . . .	EDT-22
EDT.4	Nokeypad Editing . . . . .	EDT-53
EDT.4.1	Text Entities . . . . .	EDT-53
EDT.4.2	Nokeypad Commands . . . . .	EDT-55

## Appendix ESC Escape Sequences

ESC.1	Moving the Cursor . . . . .	ESC-2
ESC.2	Setting Character Sets and Characteristics . . . . .	ESC-2
ESC.2.1	Specifying Character Sets . . . . .	ESC-3
ESC.2.2	Specifying Display Characteristics . . . . .	ESC-6
ESC.3	Erasing the Screen . . . . .	ESC-6
ESC.4	Creating a Scrolling Region . . . . .	ESC-7
ESC.5	Setting Terminal Characteristics . . . . .	ESC-7
ESC.6	Setting Keypad Characteristics . . . . .	ESC-8

## Appendix EXP Expressions

## Appendix KEY Terminal Keys

KEY.1	VT200 Terminal Series . . . . .	KEY-1
KEY.2	VT100 Terminal Series . . . . .	KEY-2

**Appendix LEX    Lexical Functions****Appendix MAIL    MAIL**

MAIL.1	DCL Command . . . . .	MAIL-1
MAIL.2	Interactive Mail Utility . . . . .	MAIL-3
MAIL.2.1	MAIL Commands . . . . .	MAIL-3
MAIL.3	MAIL Keypad . . . . .	MAIL-35
MAIL.3.1	MAIL Keypad Diagram . . . . .	MAIL-35
MAIL.3.2	MAIL Keypad Commands . . . . .	MAIL-36

**Index**



# Preface

This manual is an overview of the general use of the VAX/VMS operating system. It contains information on: logging into the system and using the terminal; using DCL, a command language that gives you interactive and batch access to the system; using MAIL, an electronic mail utility; using the file system and disk devices; using EDT, the default screen editor; writing command procedures, special files that combine DCL commands and provide programlike capabilities; and formatting text with DIGITAL Standard Runoff. For the most part, the chapters provide guidelines for performing various tasks, while the appendixes contain the component specifications. Most of the information found in the appendixes can also be obtained from the system with the HELP command and can be found in outline form in the *VAX/VMS Mini-Reference*.

## Conventions Used in This Document

Conventions	Meaning
RETURN key	The RETURN key is not shown in formats and examples. Assume that you must press RETURN after typing a command or other input to the system unless instructed otherwise.
CTRL key	The word CTRL followed by a slash followed by a letter means that you must type the letter while holding down the CTRL key. For example, CTRL/B means hold down the CTRL key and type the letter B.
Lists	When a format item is followed by a comma and an ellipsis (...), you can enter a single item or a number of those items separated by commas. When a format item is followed by a plus sign and an ellipsis (+...), you can enter a single item or a number of those items connected by plus signs. If you enter a list (more than one item), you must enclose the list in parentheses. A single item need not be enclosed in parentheses.
Optional items	An item enclosed in square brackets ([ ]) is optional.

Conventions	Meaning
Key Symbols	In examples, keys and key sequences appear as symbols, such as <code>PF2</code> and <code>CTRL/Z</code> .
Ellipses	A vertical ellipsis indicates that part of the format or example is missing.
Delete Key	The key on the VT200 series terminal keyboard that performs the DELETE function is labeled <code>&lt;X&gt;</code> . Assume that DELETE in text and examples refers to both the VT100 and VT200 series delete keys.
Examples	Examples show both system output (prompts, messages, and displays) and user input. User input is printed in red.

# Chapter 1

## Interaction with the System

VAX/VMS is an interactive system. While you are logged in, you and the system conduct a dialogue: you enter a command, the system responds, you respond, and so on.

### 1.1 Logging in to the System

To log in to your system:

1. Turn on your terminal.
2. Press the RETURN key.

You will be prompted for your USERNAME. Enter your user name and press RETURN. You will then be prompted for your PASSWORD as follows:

PASSWORD:

Type your password and press RETURN. The password is not shown on the screen. If the password is correct, you are logged in (unless a second password has been set, in which case another PASSWORD prompt appears). If the password is incorrect, you receive the message "User authorization failure" and are not logged in.

You can establish, change, or delete a password using the SET PASSWORD command, as shown in the following example. The passwords you type are not shown on the screen.

**\$ SET PASSWORD**

Old password:

New password:

Verification:

To establish a password, press RETURN when prompted for the old password, and then enter the new password when prompted for it. To delete a password, enter the old password at the prompt for the old password, and then press RETURN when prompted for the new password and verification.

## 1-2 Interaction with the System

If none of these prompts (\$, PASSWORD:, or USERNAME:) appears when you press RETURN, a system password may be required to log in to your system. If you know the system password, type it and press RETURN. If you do not know it, see the person in charge of your system.

If you are using a turnkey, or captive, account, you are running a user program, not the complete VAX/VMS operating system. Typically, you cannot reset the password or use CTRL/Y from a turnkey account.

When the DCL prompt appears, you are logged in to the system at DCL command level. Your current default device and directory are the login defaults established by the system manager. By default, the DCL prompt is a dollar sign (\$).

When you log in, the VAX/VMS system searches for your login command procedure. (A command procedure is a file containing DCL commands that are executed when the file is invoked; see Chapter 6.) Typically, your login command procedure is in a file named LOGIN.COM in your default directory. If a login command procedure exists, the system executes it before displaying the DCL prompt. Your login command procedure should contain any DCL command that you want executed each time you log in to the system.

### 1.1.1 Automatic Login

Automatic login occurs when a terminal is associated with an account. To log in on a terminal that is associated with an account, turn on the terminal and press the RETURN key. Either the DCL or password prompt appears. If the DCL prompt appears, you are logged in. If the password prompt appears, type the password of the account associated with the terminal and press the RETURN key (the password is not shown on the screen). You are logged in to the system at DCL command level. Your current default device and directory are the login defaults for the account (typically, a top-level directory on DISK1:). With automatic login enabled, the use of the terminal is restricted to the associated account.

### 1.1.2 Dialing in

Dialing in allows you to communicate with your system by telephone. To dial in to your system, you need the following items:

- Modem (or data set)—The modem is a piece of hardware that is independent of the VAX/VMS system. The user's manual that comes with the modem should describe how to connect the modem to a telephone line and a terminal.
- Terminal—You cannot dial in unless the baud rate of the terminal agrees with the baud rate of the modem and the modem terminal characteristic is set. To set the baud rate for a VT200 series terminal, select the "Comm" category in the Set-Up Directory. To set the baud rate for a VT100 series terminal, use SET-UP



B. To set the modem characteristic, use the DCL command SET TERMINAL /MODEM. If your terminal has the MODEM characteristic set (the DCL command SHOW TERMINAL lists the terminal characteristics set for your terminal), typing the SET TERMINAL/NOMODEM command causes the VAX/VMS operating system to log you out.

- Manual login account—If your account is set up for automatic login, you cannot dial in to it. Either change the account to a manual login account (one where you must type your user name) or use a different account.
- Telephone number for the system—To dial in, you must know the telephone number for your system. Ask the system manager if you do not know the telephone number.

Once the terminal is receiving the signal through the telephone line, follow the conventions your site has instituted for remote login. When communication is established, your system should respond with the prompt appropriate to your account. Note that passwords should be required for all dial-up accounts to ensure security; that is, you should not allow dial-up accounts with null passwords.

### 1.1.3 Logging in over the Network

Once you are logged in to your system, you can use the SHOW NETWORK command to determine whether your system is set up as a nonrouting node or a routing node.

- Nonrouting (end) node—A nonrouting node can receive data addressed to itself and send data to other nodes. End nodes do not send or receive information about network configurations.
- Routing node—A routing node can receive data addressed to itself and send data to other nodes. In addition, routers can receive data addressed to other nodes and forward that data to the appropriate node. A router receives information about network configurations and maintains a database to keep track of all nodes in the network.

To obtain a list of the remote systems to which you have access, use the SHOW NETWORK command. If your system is a router, the SHOW NETWORK display lists the nodes.

## 1-4 Interaction with the System

### \$ SHOW NETWORK

VAX/VMS Network status for local node 161 BEAR on 14-NOV-1986 10:05:51

Node	Links	Cost	Hops	Next Hop to Node
160 ALPHA	0	0	0	(Local) -> 160 ALPHA
161 BEAR	0	8	1	UNA-0 -> 161 BEAR
21 CUP	0	18	2	UNA-0 -> 21 CUP
41 DAN	0	18	2	UNA-0 -> 41 DAN
1 EBONY	0	18	2	UNA-0 -> 1 EBONY

Total of 6 nodes.

If your system is a nonrouter, the SHOW NETWORK display does not contain the network information.

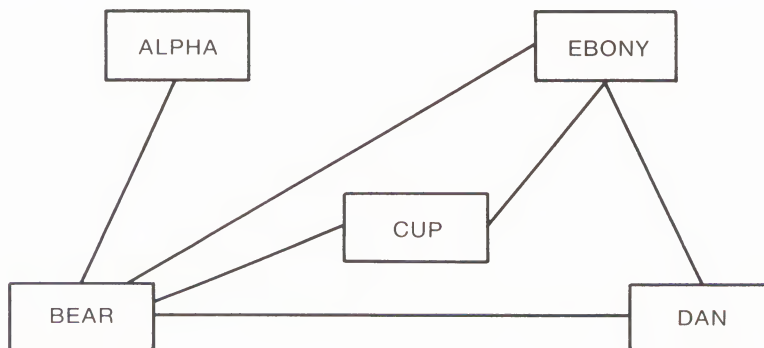
### \$ SHOW NETWORK

VAX/VMS Network status for local node 160 ALPHA on 14-NOV-1986 10:06

This is a nonrouting node, and does not have any network information.  
The designated router for ALPHA is node 161 BEAR.

If you have an account on the router system, you can log in and use the SHOW NETWORK command to examine the network. Otherwise, you must find out about your network by asking the person in charge of your system.

Each system (node) in a network is linked to one or more remote nodes, as shown:



ZK-1741-84

### 1.1.3.1 Access and Login

To access a remote node, use the SET HOST command. If the network link cannot be established, you receive an error message. Otherwise, you can log in using the remote system's login procedure. The following example assumes that you are accessing a VAX/VMS system.

```
$ SET HOST EBONY
USERNAME: RIGBY
PASSWORD:
$
```

If you attempt to gain access using invalid access information, the host system responds with the message "User authorization failure." If you want to try accessing the remote node again, press RETURN and you will again be prompted for a user name and password. If you want to abort the procedure, enter CTRL/Y twice as described in Section 1.1.3.2.

For more convenient access to frequently used remote nodes, you can create a command procedure for logging in. Invoking the following command procedure accesses the USER account on the system named EBONY.

```
$ ! EBONY.COM
$ !
$ ! Log into EBONY::USER
$ SET HOST EBONY
USER
```

If the USER account has an associated password, the password prompt appears. For security reasons, you should not put passwords into command procedures or any other files.

To further simplify access to the remote node, define a symbol to invoke the command procedure. The following symbol definition, which can be placed in your login command procedure, allows you to access the USER account on EBONY by typing EBONY (EBONY.COM is located in DISK1:[USER]):

```
$ EBONY == "@DISK1:[USER]EBONY"
```

Once logged in on a remote system, you can use that system's resources to log in to another remote node.

The HOPS entry of the SHOW NETWORK display gives the number of nodes needed to carry your data to the remote node listed. For example, the SHOW NETWORK display typed on node ALPHA (see Section 1.1.3) shows one hop for node BEAR and two hops for each of the other nodes. Multiple hops does not mean that you must log in to each intermediate node. DECnet-VAX will make all the connections for you. However, the data you type and the responses of the remote system must pass through each node used to establish the link. Therefore, a large HOPS entry (greater than 3) means a slow response from the remote system.

## 1-6 Interaction with the System

### 1.1.3.2 Terminating a Remote Session

You can terminate a remote session in two ways:

- Normally—Use the remote system's logout procedure (for example, on a VAX/VMS system, use the LOGOUT command).
- Abnormally—Press CTRL/Y twice. The host system should respond with the question, "Are you repeating ^Y to abort the remote session?" Answering Y (uppercase or lowercase) aborts the remote session. This method works regardless of the system running on the remote node.

When you terminate a remote session, the message "%REM-S-END, control returned to node \_NODENAME::" is displayed and you are returned to the calling system.

If DECnet-VAX has made intermediate connections for you and one of the intermediate systems goes down, DECnet-VAX either attempts to reroute the connection or waits for a few seconds to determine whether the system will recover. If DECnet-VAX is able to recover the connection, the interruption may be brief enough to occur without your noticing it or it may last as long as 60 seconds. If DECnet-VAX cannot recover the connection, the remote session is terminated and the message "Path lost to partner" is displayed.

### 1.1.4 Logging out of the System

When you finish using the system, you should log out. To log out, enter the LOGOUT command.

**\$ LOGOUT**

USER logged out at 14-DEC-1986 12:42:48.12

To find out how much time you spent at the terminal (elapsed time), how much computer time you used (charged CPU time), and other accounting information, you can include the /FULL qualifier to the LOGOUT command.

**\$ LOGOUT/FULL**

USER logged out at 14-DEC-1986 12:42:48.12

Accounting information:

Buffered I/O count:	8005	Peak working set size:	212
Direct I/O count:	504	Peak virtual size:	770
Page faults:	1476	Mounted volumes:	0
Charged CPU time:	0 00:00:50.01	Elapsed time:	0 02:27:43.06

## 1.2 Using the Terminal

On an interactive terminal, you send information to the system using the terminal keyboard, and the system displays information for you using the terminal screen. If you are using a turnkey account, a user program processes the data you enter rather than allowing the VAX/VMS operating system to do the processing. The following discussion assumes that you are using the VAX/VMS system.

### 1.2.1 Entering Text

Pressing a key, or combination of keys (plus RETURN) on the keyboard sends the ASCII value of the key(s) to the system. The system responds to data keys (letters, numbers, and punctuation) by echoing (displaying the characters that are typed) and storing the characters. Echoing does not occur as a direct result of your pressing the key (unless you are in local mode), but as a result of the system writing the character to your terminal. You can turn echoing off by entering the command SET TERMINAL/NOECHO and turn it back on by entering SET TERMINAL/ECHO.

Enter text through the data keys on the main keyboard and, if the SET TERMINAL/NUMERIC\_KEYPAD command is in effect (rather than SET TERMINAL/APPLICATION\_KEYPAD), through the numeric keypad to the right (excluding the PFn keys). If the SET TERMINAL/APPLICATION\_KEYPAD command is in effect, you can define the keys of the numeric keypad for other uses, but you cannot use them to enter numbers and punctuation marks.

The data keys are supplemented by several keys, including SHIFT and RETURN. Keys that allow you to edit what you are typing are described in Section 1.4.7. Control keys (entered by pressing the key labeled CTRL and, at the same time, a data key), are referred to as CTRL/x, where x is the data key. Control keys are described in sections pertaining to their functions; see Appendix KEY for a complete list of control keys. (To pass a control character to an image rather than allowing the system to process the character, press CTRL/V and then enter the control character.)

Function keys on the VT200 terminal are referred to as Fx, in which x is the number associated with a particular function key. The function keys are located above the main keyboard and are labeled F1 through F20. Function keys are described in sections pertaining to their use; see Appendix KEY for a complete list of function keys.

If the system is not ready for your input when you enter it at the terminal, it is stored in a special area called the type-ahead buffer. When the system is ready, it processes your input from the type-ahead buffer and echoes it. In this way, you can enter input before the system is ready to receive it without losing it. You can disable the type-ahead buffer with the SET TERMINAL/NOTYPE\_AHEAD command. Your input is processed immediately, and if the system is not ready for the input, it is lost.



## 1-8 Interaction with the System

You can reenable the type-ahead buffer with the SET TERMINAL/TYPE\_AHEAD command. Note that certain control characters (such as CTRL/Y and CTRL/O) are not put in the type-ahead buffer; they are processed immediately.

### 1.2.2 Controlling Output

You can control the formatting and movement of output on your terminal.

#### 1.2.2.1 Suspending/Resuming Terminal Display

You may need to stop a lengthy screen display so that you can read it before it disappears off the top of the screen. Use the F1 key on VT200 terminals (the NO SCROLL key on VT100), or CTRL/S, to stop output to the screen. To resume output, press the F1 (NO SCROLL) key a second time, or press CTRL/Q. The display continues from the point where you halted it. The F1 (NO SCROLL) key works only if AUTO XON/XOFF is enabled; see Section 1.3.1.3.

If you want to skip over portions of the text being displayed, use CTRL/O. Pressing CTRL/O does not suspend output but suspends the echoing of the output. Therefore, when you press CTRL/O a second time, echoing resumes at the current position in the text rather than from the point where you suspended the text.

#### 1.2.2.2 Formatting Screen Output with Escape Sequences

An escape sequence is a code consisting of an escape character followed by one or more data characters. Escape sequences allow you to send formatting and other control commands to the terminal. However, if escape sequences are sent to a device that does not support them (for instance, a hardcopy terminal), the results are unpredictable. Escape sequences must be specified exactly as described; case (uppercase or lowercase) and punctuation are significant. Appendix ESC describes the VT200 and VT100 escape sequences.

You can send escape sequences to a terminal in three ways:

- **Command procedure**—Use an editor to create a command procedure that equates the low-order 8 bits of a symbol to 27, the ASCII value of the ESC character, and then writes escape sequences and text to the terminal. Use the symbol rather than the escape character itself so that you can examine the file if you accidentally send it to a device that does not support escape characters. When executed, the following command procedure writes the word BOX in bold characters inside a box drawn with special graphic characters (Appendix ESC describes the graphic character set). The three apostrophes enclosing the symbol ESC force the VAX/VMS system to translate the symbol; see Section 5.5.3 for more information about symbol substitution.

```

$ ! BOX.COM
$ !
$ ! define the escape character
$ ESC[0,8] = %X1B
$
$ WRITE SYS$OUTPUT "'ESC'(0"
$ WRITE SYS$OUTPUT "'ESC'[10;35Hlqqqqqk"
$ WRITE SYS$OUTPUT "'ESC'[11;35Hx'ESC'[11;41Hx"
$ WRITE SYS$OUTPUT "'ESC'[12;35Hmqqqqqj"
$ WRITE SYS$OUTPUT "'ESC'(B"
$ WRITE SYS$OUTPUT "'ESC'[11;37H'ESC'[7mBOX"
$ WRITE SYS$OUTPUT "'ESC'[0m"

```

In this example, the escape sequences were written in different statements for clarity. However, lists of escape sequences can be on one or multiple lines, provided that no escape sequence is split between two lines.

- **Text file**—Use an editor to create a file containing escape sequences and text. (Insert an escape character into the text by pressing CTRL/[ on VT200 terminals or the ESC key on VT100 terminals twice.) Use the TYPE command to send the file from the system to the terminal. Since a file containing escape characters could mistakenly be sent to a device that does not support escape sequences, this technique is not generally recommended.
- **LOCAL mode**—Use the set-up feature (see Section 1.3) to put your terminal in LOCAL mode. In local mode, characters are sent directly to the screen, not to the system. For example, if you put the terminal in LOCAL mode, enter the escape character CTRL/[ and then type [2], the screen is erased. The escape sequence <ESC> [2] is not echoed.

Most commonly, escape sequences are used to enhance the appearance of the text when it is displayed or to make use of the graphic character set.

The terminal attribute Escape determines how the VAX/VMS system interprets the escape character; it does not affect the execution of escape sequences. If Escape is enabled, the system recognizes the escape character as the beginning of an escape sequence, waits for the control characters that should follow, and interprets the entire sequence as a line terminator. If Escape is disabled, the system recognizes the escape character as a line terminator and echoes a dollar sign to the terminal. Since a program can override the Escape attribute, even when you have disabled Escape it may appear to be enabled. (Enable and disable Escape with the /[NO]ESCAPE qualifier of the SET TERMINAL command.)

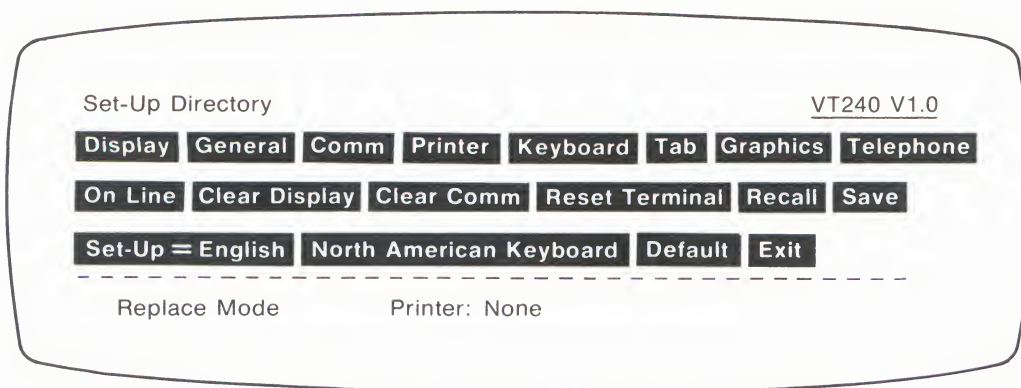
## 1.3 Setting the Terminal's Physical Attributes

You can set the physical attributes for your terminal by using the set-up feature for your particular terminal. The DCL command SET TERMINAL sets the logical attributes for your terminal and affects some of its physical attributes. See Appendix DCL for a description of the SET TERMINAL command.

### 1.3.1 VT200 Series

Set-up on the VT200 series terminal is based on several set-up screens that you can select from the Set-Up Directory. To enter Set-Up, press the F3 keyboard key; the Set-Up Directory will be displayed on the lower third of your terminal screen.

To set physical attributes on the VT200 series terminal, you move the cursor (displayed in reverse video) from field to field using the arrow keys.



ZK-1683-84

To exit from Set-Up, either press the F3 key or select the Exit field in the Set-Up Directory (as explained in Section 1.3.1.2) and then press the ENTER keypad key.

Each set-up screen contains a screen title, a terminal identifier, a firmware version number, a status line, and attribute fields. To modify an attribute:

1. Position the cursor—Use the arrow keys to position the cursor over the field containing the attribute you want to change.
2. Press the ENTER keypad key—The ENTER key changes the setting of each field.



There are three types of fields, each of which responds differently when the ENTER key is pressed.

Action field	An action field has only one value and is activated by pressing the ENTER key.
Parameter field	A parameter field has two or more values. The next value is selected by pressing the ENTER key.
Text parameter field	In text parameter fields, direct entry of text is accomplished by typing the text or number you want entered on the status line at the bottom of your screen. Pressing the ENTER key completes the operation.

### 1.3.1.1 The Set-Up Directory

The top row in the Set-Up Directory consists of eight action fields. If you move the cursor to one of these action fields and press ENTER, another set-up screen (Display, General, Comm, Printer, Keyboard, Tab, Graphics, or Telephone) will be displayed on your terminal.

**NOTE:** The Telephone field only appears in the Set-Up Directory screen when the VT200 series Integral/Modem option is installed.

The second row contains one parameter field and five action fields.

- On Line/Off Line (parameter) field—Determines whether your terminal is connected to the VAX/VMS operating system or whether data entered is sent directly to the terminal screen.
- Clear Display (action) field—Clears the terminal screen.
- Clear Comm (action) field—Clears the communication lines; aborts any escape sequence, control sequence, or DCS processing; clears the keyboard buffers; clears the receive buffer; sends XON to the host port; resets XOFF received flags on both the printer and host ports; and takes the printer out of controller mode.
- Reset Terminal (action) field—Resets many internal terminal features; however, this action does not affect communication and user-defined keys.
- Recall (action) field—Replaces all the existing set-up characteristics with the saved values and clears the terminal screen. Note that Recall causes a disconnect to occur.
- Save (action) field—Saves the current set-up features from the Set-up screen.

The third row of the Set-Up Directory contains two parameter fields and two action fields.

- Set-Up=[language] (parameter) field—Allows you to choose the language in which you want the Set-Up attributes displayed. This field takes effect immediately.

## 1-12 Interaction with the System

- [National] Keyboard (parameter) field—Allows you to select correct terminal operation for the national keyboard you are using.
- Default (action) field—Replaces all current set-up features with the factory default settings. The terminal screen is cleared and the cursor is returned to the upper left corner of the screen. Note that Default causes a disconnect to occur.
- Exit (action) field—Exits from Set-Up and returns the terminal to On-Line or Local, depending on which feature is set.

### 1.3.1.2 Changing Attributes with Set-Up

To change a terminal attribute that is in the Set-Up Directory, select the attribute field with the arrow keys. To change an attribute that is located in another screen, select the appropriate screen and press ENTER. When the selected screen appears, select the appropriate field. The following table lists the attributes available in the Set-Up Directory and the eight other set-up screens:

Set-Up Directory	Display Set-Up	General Set-Up
Display Set-Up	To Next Set-Up	To Next Set-Up
General Set-Up	To Directory	To Directory
Communications Set-Up	80/132 Columns	Terminal Mode
Printer Set-Up	Interpret/Display Controls	VT100 ASCII/UK
Keyboard Set-Up	Auto Wrap	UDK Lock
Tab Set-Up	Smooth/Jump Scroll	User Features Lock
Graphics Set-Up	Light/Dark Text/Screen	Keypad Mode
Telephone Set-Up	Display Select	Cursor Key Mode
On Line/Local	Text Cursor	New Line
Clear Display	Block/Underline Cursor	
Clear Communications		
Reset Terminal		
Recall Saved Attributes		
Save Attributes		
Set-Up Language		
Keyboard Language		
Set Factory Defaults		
Exit Set-Up		

<b>Communications Set-Up</b>	<b>Printer Set-Up</b>	<b>Keyboard Set-Up</b>
To Next Set-Up	To Next Set-Up	To Next Set-Up
To Directory	To Directory	To Directory
Transmit Speed	Transmit/Receive Speed	Typewriter/D.P.
Receive Speed	Printer to Host Mode	Caps/Shift Lock
XOFF	Print Mode	Auto Repeat
Data-Bits Parity	XOFF	Keyclick
Stop Bits	Data-Bits Parity	Margin Bell
Local Echo	Stop Bits	Warning Bell
Host Port Selection	Text Print Extent	Break
Disconnect	Printed Data Type	Auto Answerback
Transmit Rate Limit	Print Terminator	Answerback Message
		Conceal Answerback

<b>Tab Set-Up</b>	<b>Graphics Set-Up</b>	<b>Telephone Set-Up</b>
To Next Set-Up	To Next Set-Up	To Next Set-Up
To Directory	To Directory	To Directory
Clear All Tabs	4010 CR Effect	Auto Answer
Set 8 Column Tabs	4010 Line Feed Effect	Telephone Number A
Tab Fields and Ruler	4010 GIN Terminator	Conceal A
	4010 DEL as Lo Y	Telephone Number B
	Macrograph Lock	Conceal B
	Graphics Cursor	
	Compressed/Expanded Print	

For complete information about your VT200 series terminal and its set-up features, refer to your terminal owner's manual.

For example, to change the background color of the screen, enter the Set-Up Directory by pressing the F3 key. Use the arrow keys to select the Display field (this field should be the current field when you enter the Set-Up Directory), and press the ENTER key. Once in the Display Set-Up screen, select the Light Text, Dark Screen/Dark Text, Light Screen field (third field in the second row), and press the ENTER key. The screen responds immediately by setting the text and screen tones.

By default, all the changes you make are temporary—the attributes remain until you reset the terminal or turn it off. To make the changes permanent, select the Save field in the Set-Up Directory and then press the ENTER key. The word "Done" will appear on the status line at the bottom of the Set-Up screen indicating that your terminal attributes have been saved.

## 1-14 Interaction with the System

### 1.3.1.3 Critical Attributes

Make sure the following critical attributes are set correctly:

- **Baud Rate**—In the Communications Set-Up screen are two fields shown as Transmit= \_ \_ \_ \_ and Receive= \_ \_ \_ \_ . These fields indicate the transmit and receive baud rates for the terminal. Move the cursor to either of these fields and press the ENTER key repeatedly to move through the available baud rates in the following order: 75, 110, 150, 300, 600, 1200, 2400, 4800, 9600, 19,200. Both the Transmit= \_ \_ \_ \_ and the Receive= \_ \_ \_ \_ fields should be set to 9600 if your terminal is directly connected to the VAX/VMS system. Dial-in terminals are usually set at 1200 or 300 baud.
- **XOFF Point**—Also located in the Communications Set-Up screen is an XOFF field. This field determines the XOFF point or disables the automatic XON/XOFF flow control. Move the cursor to the XOFF field (first field in the second row), and press the ENTER key to move through the XOFF values in the following order: XOFF at 64, XOFF at 256, XOFF at 512, XOFF at 1024, No XOFF. The XOFF field should be set to XOFF at 64.

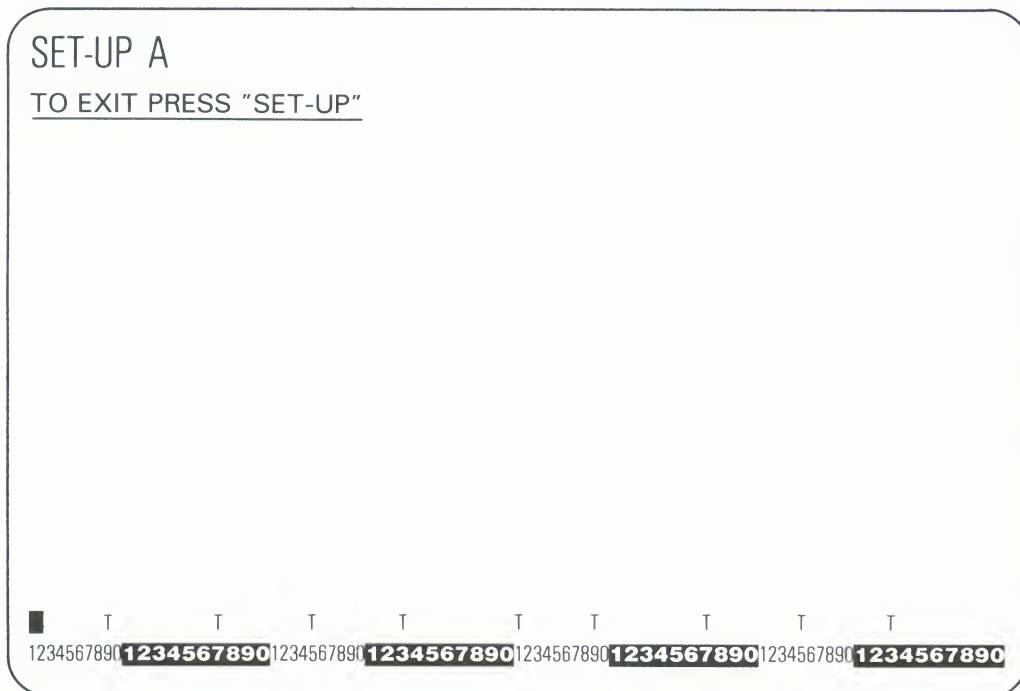
### 1.3.2 VT100 Series

To change the physical attributes of a terminal in the VT100 series, enter SET-UP mode by pressing the SET-UP key. The terminal display indicates that you are in SET-UP mode A. To enter SET-UP mode B, press the SET-UP A/B key (the key labeled 5 at the top of the keyboard). To exit from SET-UP mode, press the SET-UP key.

By default, the changes you make are temporary—the attributes remain set until you reset the terminal or turn it off. To make the changes permanent, press SHIFT/S before exiting from SET-UP mode. To set the terminal to your permanent attributes, enter SET-UP mode (either A or B) and press the RESET key (number 0).

**1.3.2.1 SET-UP Mode A**

Pressing the SET-UP key invokes the SET-UP A display:



ZK-1742-84

Use SET-UP mode A to change the following attributes:

- **Columns**—The numbers across the bottom of the screen in SET-UP mode A mark the columns. The screen can be either 80 or 132 columns wide. To change the number of columns on the screen, press the 80/132 COLUMNS key (number 9): if the screen held 80 columns, it now holds 132; if the screen held 132 columns, it now holds 80. Changing the number of columns on the screen does not change the number of characters the system places on a line. Use the DCL command SET TERMINAL/WIDTH to specify the number of characters that the system writes on a single line.
- **Tabs**—You can set a tab in any column on the terminal screen. A tab is indicated by a "T" placed above the column number at the bottom of the screen in SET-UP mode A. To set or clear a tab, use the left and right arrow keys to position the cursor in the proper column and press the SET/CLEAR TAB key (number 2). If the column already has a tab, pressing the SET/CLEAR TAB key clears it; if the

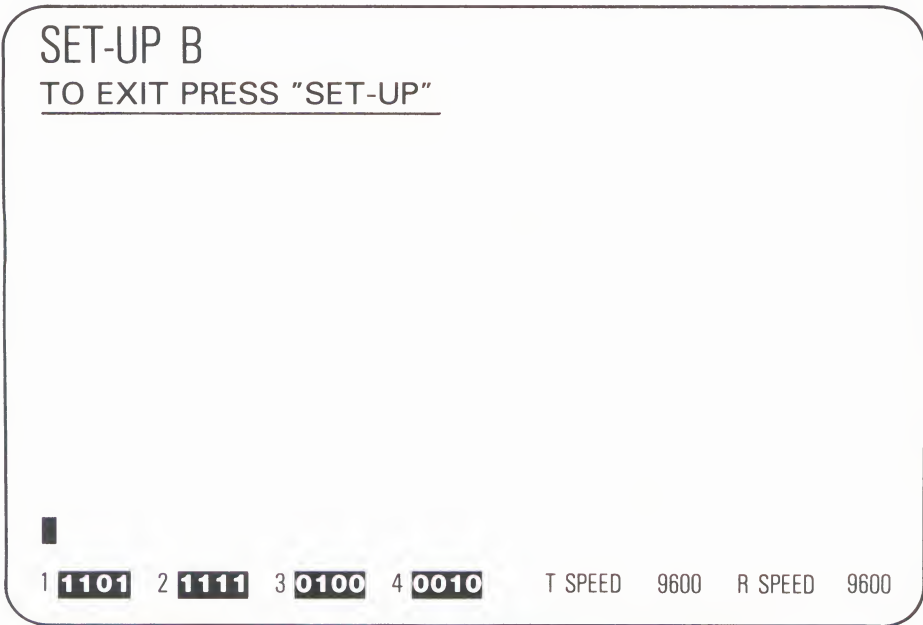
1-16    Interaction with the System

column does not have a tab, pressing the SET/CLEAR TAB key sets one. To clear all the tabs, press the CLEAR ALL TABS key (number 3).

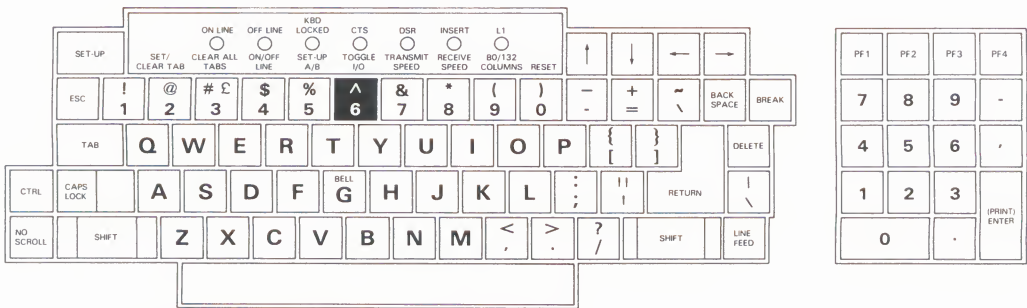
The tabs you set in SET-UP can be used to format information in files created with the DCL command CREATE. However, if you use an interactive editor, you must use the tab commands provided by the editor since interactive editors ignore the tabs set in SET-UP.

1.3.2.2 SET-UP Mode B

To invoke SET-UP B, first press the SET-UP key and then the SET-UP A/B key (the key labeled 5 at the top of the keyboard):



ZK-1687-84



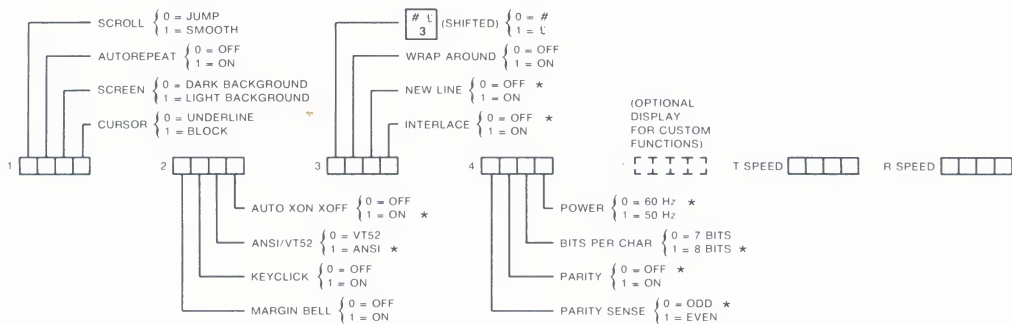


Each of the 1s and 0s contained in the four boxes on the lower left corner of the display is associated with an attribute, as shown in the following figure. To change one of these attributes:

1. Position the cursor—Use the left and right arrow keys to position the cursor over the number associated with the attribute you want to change.
2. Press the TOGGLE 1/0 key (number 6)—The number below the cursor changes: if the number is 0, it becomes 1; if the number is 1, it becomes 0.

For example, to change the background color of the screen, enter SET-UP mode B, position the cursor over the third number in the first box, and press the TOGGLE 1/0 key (number 6).

A number of the attributes you can change in SET-UP mode B affect the operation of the terminal. In the following figure, the proper settings for these attributes are marked with an asterisk.



2/8 1762 84

You should make sure that the baud rates for your terminal are set correctly. In the lower right corner of the SET-UP mode B display are two numbers labeled T SPEED and R SPEED. These numbers indicate the transmit and receive baud rates. Use the TRANSMIT SPEED and RECEIVE SPEED keys (numbers 7 and 8, respectively) to set these rates. Pressing either TRANSMIT SPEED or RECEIVE SPEED causes the terminal to move through the baud rates in the following order: 50, 75, 110, 134.5, 150, 200, 300, 600, 1200, 1800, 2000, 2400, 3600, 4800, 9600, 19200. If your terminal is connected directly to the VAX/VMS system, both the transmit and receive speeds should be set at 9600. (Some modems require a different baud rate; see Section 1.1.2.)

## 1.4 Commands and Utilities

The DIGITAL command language (DCL) provides you with a direct connection to the VAX/VMS system and the software running on VAX/VMS. In response to the DCL prompt (which is initially a dollar sign), you enter a command name followed by any desired parameters and qualifiers. DCL interprets the command, and either executes it directly (a so-called "built-in" command) or calls an appropriate program to execute it, passing to that program any parameter and qualifier information. (Most of the commands not given in the list of built-in commands are in this category.)

Some DCL commands invoke utilities that themselves accept interactive subcommands. You then work interactively with the program by entering subcommands and other information in response to the utility's command prompts. You continue to work with the utility until you exit from it and return to DCL command level. The MAIL command falls into this category.

A program that is associated with a command (a command image) can be DIGITAL- or user-supplied. The built-in commands and the commands that execute system programs are supplied and supported by DIGITAL as part of the operating system. Appendix DCL contains complete descriptions of each of these commands.

Built-in commands:

=, ==, :=, :==	ALLOCATE	ASSIGN
ATTACH	CALL	CANCEL
CLOSE	CONNECT	CONTINUE
CREATE/LOGICAL_NAME_TABLE	DEALLOCATE	DEASSIGN
DEBUG	DECK	DEFINE
DEFINE/KEY	DELETE/KEY	DELETE/SYMBOL
DEPOSIT	DISCONNECT	ENDSUBROUTINE
EOD	EXAMINE	EXIT
GOSUB	GOTO	IF
INQUIRE	ON	OPEN
READ	RECALL	RETURN
SET CONTROL	SET DEFAULT	SET KEY
SET ON	SET OUTPUT_RATE	SET PROMPT
SET PROTECTION/DEFAULT	SET UIC	SET VERIFY
SHOW DEFAULT	SHOW KEY	SHOW PROTECTION



SHOW QUOTA	SHOW STATUS	SHOW SYMBOL
SHOW TIME	SHOW TRANSLATION	SPAWN
SHOW TIME	SUBROUTINE	WAIT
WRITE		

A utility is a program that provides some user service; it refers both to interactive commands and to some complex noninteractive commands.

### 1.4.1 DCL Command Format

A DCL command follows the general format:

`command [/command-qualifier...] [parameter[/parameter-qualifier...]]...`

where *command* is the name of the command, *command-qualifier* is the name of a command qualifier, *parameter* is the name of a parameter, and *parameter-qualifier* is the name of a parameter qualifier. Lowercase and uppercase characters in command and qualifier names are equivalent. Lowercase and uppercase characters in parameter and parameter qualifier values are equivalent unless enclosed in quotation marks. The following command line consists of the command name, a command qualifier, and two parameters.

```
$ COPY/LOG FORMAT.TXT WATER.TXT
```

The command instructs the system to copy the file FORMAT.TXT to another file named WATER.TXT and display (log) the status of the operation. (To implement the command, DCL interprets the command and calls the system program SYS\$SYSTEM:COPY.EXE, passing it the qualifier and parameter information.) You must observe the following rules in entering DCL commands:

- Delimiters—Delimit the command name and parameters with one or more blanks or tabs or qualifiers. Begin each qualifier with a slash (/); the slash serves as a delimiter and need not be preceded by blanks or tabs.
- Continuation over many lines—You can continue a command line by terminating it with a hyphen, pressing RETURN, and entering more of the command on the next line (although a single command line can not exceed 256 characters), as demonstrated:

```
$ COPY/LOG FORMAT.TXT,WATER.TXT,SOIL.TXT -
_ $ SAVE.TXT
```

The system responds to the hyphen and RETURN with the prompt string preceded by an underscore. Note that the space delimiting command names and parameters must be supplied (the RETURN is not treated as a delimiter).

- **Size limit**—An element in a command (for example, a qualifier and associated values) must not exceed 255 characters. The number of elements in a command must not exceed 128. The entire command must not exceed 1024 characters after all symbols and lexical functions are converted to their values.
- **Abbreviation**—You can abbreviate a command name by truncating it if the abbreviated name is still unique among all the DCL command names. You can abbreviate a qualifier name if it remains unique among all qualifier names for the same command. (For clarity, the examples in this manual do not abbreviate commands or qualifiers.) All command and qualifier names are unique within four characters (not counting the slash before qualifiers). The following commands, for example, are equivalent:

```
$ PR/C=2 WATER.TXT
$ PRINT/COPIES=2 WATER.TXT
```

In interactive mode, you will work faster if you abbreviate. The abbreviations may feel awkward at first but you will soon get used to them. You should not abbreviate commands in command procedures because: (1) your command procedure will be difficult to read, and (2) the abbreviations might not be valid after new DCL commands are added at a later date.

Other rules governing the format of commands apply mainly to their use in command procedures (see Chapter 6).

## 1.4.2 Parameters

A parameter consists of a value or a list of values. You must position parameters in a specified order within the command, as demonstrated:

```
$ COPY WATER.TXT FORMAT.TXT
```

This command causes the file WATER.TXT to be copied to FORMAT.TXT. The following command reverses the order of the parameters, copying the file FORMAT.TXT to WATER.TXT.

```
$ COPY FORMAT.TXT WATER.TXT
```

Specify a parameter list by separating the values with commas (in some commands, you can use plus signs to denote concatenation of files). The following example copies a number of files to another file.

```
$ COPY FORMAT.TXT,WATER.TXT,SOIL.TXT SAVE.TXT
```

The entire list FORMAT.TXT,WATER.TXT,SOIL.TXT constitutes the first parameter. SAVE.TXT is the second parameter.

### 1.4.3 Qualifiers

A qualifier consists of a keyword, or a keyword followed by a value or list of values. The keyword starts with a slash. Three general classes of qualifiers exist:

- **Command qualifiers**—A command qualifier applies to the entire command; the best practice is to place it after the command name (or after other command qualifiers following the command name), although in some cases a command qualifier can appear anywhere in the command. The following example prints two copies each of WATER.TXT and SOIL.TXT:  

```
$ PRINT/COPIES=2 WATER.TXT,SOIL.TXT
```
- **Positional qualifiers**—A positional qualifier has different meanings depending on where you place it in the command string. If you place a positional qualifier after the command verb but before the first parameter, the qualifier affects the entire command string. If you place a positional qualifier after a parameter, the qualifier affects only that parameter. In the following example, the first PRINT command 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.  

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

- **Parameter qualifiers**—A parameter qualifier applies only to the parameter value it follows. The following example prints two copies of WATER.TXT and three copies of SOIL.TXT:

```
$ PRINT WATER.TXT/COPIES=2,SOIL.TXT/COPIES=3
```

Within the confines of the above rules, the relative position of qualifiers in a command does not matter. Qualifiers take one of the following formats:

- **Positive-negative qualifiers**—Positive-negative qualifiers have a value of true or false. You do not specify a value, but indicate a true value by simply naming the qualifier, or negate the qualifier by inserting the prefix NO. The first example that follows puts a flag page at the beginning of the print job, while the second example does not.

```
$ PRINT/FLAG_PAGE WATER.TXT,SOIL.TXT
$ PRINT/NOFLAG_PAGE WATER.TXT,SOIL.TXT
```

- **Value qualifiers**—If the qualifier accepts a value, you specify it by appending an equal sign and the value, as demonstrated:

```
$ PRINT/COPIES=2 WATER.TXT
```

The /COPIES qualifier has a value of 2.

## 1-22 Interaction with the System

- Lists of values for qualifiers—If the qualifier accepts a list of values, you must enclose the values in parentheses and separate them with commas, as demonstrated:

```
$ DELETE/ENTRY=(230,231) SYS$BATCH
```

The command deletes jobs 230 and 231 from the queue SYS\$BATCH.

- Value and positive-negative combinations—Some qualifiers combine positive-negative and value characteristics so that the qualifier accepts a value. The SET TERMINAL command, for example, permits the following choices for the /PARITY qualifier.

```
$ SET TERMINAL/PARITY=EVEN
```

```
$ SET TERMINAL/PARITY=ODD
```

```
$ SET TERMINAL/NOPARITY
```

- Defaults—Most of the qualifiers take defaults or do not affect existing values when they are not specified. For example, the following commands are equivalent because the qualifiers /KEEP=1 and /NOLOG are the defaults for the PURGE command.

```
$ PURGE [.MEMOS]
```

```
$ PURGE [.MEMOS]/KEEP=1/NOLOG
```

The following command affects only the width of your terminal screen. All the other terminal characteristics remain the same.

```
$ SET TERMINAL/WIDTH=132
```

### 1.4.4 Prompts

If you omit a required parameter from a command, the system prompts you for the parameter as demonstrated:

```
$ PRINT
```

```
_File:    WATER.TXT
```

You are also prompted for any additional parameters, required or optional.

```
$ ALLOCATE
```

```
_Device:  DL
```

```
_Log_Name: ACCOUNTS_DISK
```

You can omit optional parameters by just pressing RETURN. On any prompt, you can enter one or more of the remaining parameters and any additional qualifiers.

```
$ ALLOCATE
```

```
_Device:  DL ACCOUNTS_DISK
```

### 1.4.5 Interactive Commands

You invoke an interactive command by typing its name and pressing RETURN. (Some interactive commands accept parameters and qualifiers on the command line.) The command responds with a prompt, as shown:

```
$ MAIL
MAIL>
```

You can now enter subcommands recognized by that interactive command. To enter another DCL command, you must first exit from the interactive command, usually by typing EXIT (and pressing RETURN) or pressing CTRL/Z in response to the command prompt.

To obtain help for an interactive command, invoke the command and then type HELP (and press RETURN) as you would at DCL command level (see Section 1.4.8).

### 1.4.6 Interrupting Commands

You can interrupt the execution of a command by pressing CTRL/Y, CTRL/C, or CTRL/T (the current program may have redefined CTRL/C or CTRL/T, in which case the usual system actions are overridden). CTRL/T interrupts execution of the command, displays a line of information (node name, process name, system time, elapsed CPU time, page faults, direct and buffered I/O operations, and pages in physical memory), and resumes execution. The following example interrupts the copy operation to display CTRL/T information and then resumes the copy operation.

```
$ COPY [.MEMOS]*.* *
[CTRL/T]
BEAR::USER 16:54:17 COPY CPU=00:00:01.16 PF=241 IO=47 MEM=141
.
.
.
```

**NOTE:** You must enable system recognition of CTRL/T by using the DCL command SET CONTROL=T.

CTRL/Y interrupts a command and returns you to DCL command level without completing execution of the command.

```
$ COPY [.MEMOS]*.* *
[CTRL/Y]
INTERRUPT
$
```



After interrupting a command with CTRL/Y, you can:

- Exit normally—In a normal exit, the program executes any cleanup procedures before terminating. To terminate the program normally, enter any command that causes a program to execute or enter the EXIT command. (If the next command is a built-in command, the exit is delayed until a command executing a program is entered.)
- Exit abnormally—In an abnormal exit, the program terminates immediately. To terminate the program abnormally, enter the command STOP. (The advantage to using STOP over simply continuing with the next command is that STOP suppresses any cleanup activities—for example, the display of error messages.)
- Continue—You can continue execution of the program by entering the command CONTINUE. Any number of built-in commands (but only built-in commands) can be entered after CTRL/Y and before CONTINUE. The following example interrupts the execution of the CLEANUP command procedure, sets verification, and then continues execution of the command procedure.


```
$ @CLEANUP
  CTRL/Y
  INTERRUPT
$ SET VERIFY
$ CONTINUE
```

CTRL/C works like CTRL/Y unless the program you are executing responds to CTRL/C (which is a common practice).



### 1.4.7 Editing Command Lines

The following keys allow you to edit the current DCL command line (and the command lines of most utilities). For some of these keys to work, the SET TERMINAL/LINE\_EDITING command must be in effect. (Enter the SHOW TERMINAL command to display your terminal's attributes.)

VT200 Key	VT100 Key	Function
F12,CTRL/H	BACKSPACE,CTRL/H	Moves the cursor to the beginning of the line
F14,CTRL/A	CTRL/A	Changes between SET TERMINAL/OVERSTRIKE and SET TERMINAL/INSERT
CTRL/B	CTRL/B	Recalls up to 20 previously entered commands

VT200 Key	VT100 Key	Function
CTRL/E	CTRL/E	Moves the cursor to the end of the line
CTRL/R	CTRL/R	Repeats the current command line
CTRL/U	CTRL/U	Deletes the characters to the left of the cursor
	DELETE	Deletes the character to the left of the cursor, moving the cursor one space to the left
F13	LINEFEED	Deletes the word to the left of the cursor
DOWN ARROW	DOWN ARROW	Recalls the command entered after the current command
RIGHT ARROW,CTRL/F	RIGHT ARROW,CTRL/F	Moves the cursor one character right
LEFT ARROW,CTRL/D	LEFT ARROW,CTRL/D	Moves the cursor one character left
UP ARROW	UP ARROW	Recalls the command entered before the current command

Command-line editing is most useful for modifying long command lines. You can edit command lines that contain typographical errors or command lines that you have recalled and want to modify (see Section 1.5.4 for a description of how to recall previous command lines). For example, the following command line contains one mistake that can be corrected more easily than retyping the entire command line. (Where line-editing keys on the VT200 series and VT100 series terminals differ, VT200 keys are pictured first and VT100 keys are pictured in parentheses.)

```
$ DILETE WORKMAR1986: [ACCOUNTS]SMITH.DAT;3
       
$ DILETE WORKMAR1986: [ACCOUNTS]SMITH.DAT;3
      →
$ DILETE WORKMAR1986: [ACCOUNTS]SMITH.DAT;3
      E
$ DELETE WORKMAR1986: [ACCOUNTS]SMITH.DAT;3 RETURN
```

This example assumes that the SET TERMINAL/OVERSTRIKE attribute is in effect (as it is by default). The OVERSTRIKE attribute allows you to replace the incorrect character by typing the correct character over it (replacing I with E in the preceding example). To insert characters in the command line without simultaneously deleting them, change the OVERSTRIKE attribute to the INSERT attribute. While you

## 1-26 Interaction with the System

are editing a command line, you can set the OVERSTRIKE or INSERT attributes temporarily by entering F14 (or CTRL/A). Use the SET TERMINAL command to set the attribute permanently for your terminal.

### 1.4.8 Help

To obtain online documentation for a command, enter the command HELP with the name of the command as a parameter.

```
$ HELP ALLOCATE
```

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[:]]
```

Additional information available:

Parameters Command\_Qualifiers

/LOG /GENERIC

Examples

ALLOCATE Subtopic?

If you need help, but do not know what command or system topic to specify, enter the command HELP with the word HINTS as a parameter. Each task name listed in the HINTS text is associated with a list of related command names and system information topics.

```
$ HELP HINTS
```

HINTS

Type the name of one of the categories listed below to obtain a list of related commands and topics. To obtain detailed information on a topic, press the RETURN key until you reach the "Topic?" prompt and then type the name of the topic.

Topics that appear in all upper case are DCL commands.

Additional information available:

Batch_and_print_jobs	Command_procedures	Contacting_people
Creating_processes	Developing_programs	Executing_programs
Files_and_directories	Logical_names	Physical_devices
System_management	Terminal_environment	User_environment

HINTS Subtopic?



When HELP prompts you for a topic or subtopic, you can enter one of the listed subtopics to obtain additional information (command and topic names can be abbreviated). Alternatively, you can press RETURN to move back a level, enter a question mark to redisplay the current text, or press CTRL/Z to exit. (You can use HELP interactively by entering just the command HELP.)

Using wildcard characters when specifying a topic allows you to obtain various amounts of information.

Format	Information Provided
HELP command...	The command or topic and all related information
HELP command *	All related information
HELP com*	All commands or topics beginning with the specified character(s)

## 1.5 Shortcuts for Entering Commands

You can use symbols, command procedures, and key definitions to simplify the typing of command lines. In addition, you can use CTRL/B and the RECALL command to reissue any of your last 20 command lines.

### 1.5.1 Symbols

Use symbols to tailor commands to suit your site conventions or personal habits. For example, you can shorten the SHOW DEFAULT command by equating it to a symbol.

```
$ SD = "SHOW DEFAULT"
```

Now you can issue the SHOW DEFAULT command by typing SD. If you wish to use a symbol each time you log in, make it a global symbol (by using two equal signs) and place the symbol definition in your login command procedure.

```
$ ! LOGIN.COM
```

```
$ !
```

```
$ SD == "SHOW DEFAULT"
```

A useful device is to equate SET DEFAULT commands for your most frequently used directories to easily remembered symbols. For example, you might equate each command to the name of the directory with the letter D appended.

```
$ ! LOGIN.COM
```

```
$ !
```

```
$ ACCOUNTD == "SET DEFAULT WORKDISK: [USER.ACCOUNT]"
```

```
$ MEMOD == "SET DEFAULT WORKDISK: [USER.MEMO]"
```

### 1.5.2 Command Procedures

Use command procedures to combine commands. For example, if you like to examine the contents of a directory immediately after setting your default to it, you might set up a command procedure that issues the appropriate commands. The command procedure for the ACCOUNT subdirectory might contain the following commands.

```
$ ! WORKDISK: [USER]ACCOUNTD.COM
$ !
$ SET DEFAULT WORKDISK: [USER.ACCOUNT]
$ DIRECTORY
```

Now you type @WORKDISK:[USER]ACCOUNTD (or @ACCOUNTD if your current default is WORKDISK:[USER]) to change directories and view the contents of the new directory.

Rather than write a separate command procedure for each directory, you could write a single command procedure that would accept the name of a directory as a parameter, change your default to the specified directory, and view the contents of the new directory. In the following command procedure, the symbol P1 is equated to a value that you specify as a parameter when you invoke the command procedure.

```
$ ! WORKDISK: [USER]DIRECTORY.COM
$ !
$ SET DEFAULT 'P1'
$ DIRECTORY
```

To simplify the invocation of DIRECTORY.COM, you could equate DIRECTORY.COM invocations for each of your directories to easily remembered global symbols. For example, you might equate each invocation to the name of the directory with the letter D appended.

```
$ ! LOGIN.COM
$ !
$ ACCOUNTD == "@WORKDISK: [USER]DIRECTORY WORKDISK: [USER.ACCOUNT]"
```

Chapter 6 discusses command procedures in greater detail.

### 1.5.3 Key Definitions

Use key definitions to equate a key on your terminal to a command or partial command. To define keypad keys, issue the DCL command SET TERMINAL /APPLICATION\_KEYPAD. Then, for example, you might equate the PF1 key to the SET DEFAULT command.

```
$ DEFINE/KEY PF1 "SET DEFAULT "  
%DCL-I-DEFKEY, DEFAULT key PF1 has been defined
```

Now you can issue the SET DEFAULT command by pressing the PF1 key. When you press the PF1 key, the words SET DEFAULT are entered and echoed as if you had typed them. To complete the command, type the name of a device and/or directory and press RETURN. (If you wish to use a key definition each time you log in, place the key definition in your login command procedure. If you wish to make the key definition available to everyone using the system, place the key definition in the system login command procedure.)

Rather than typing the device/directory names after SET DEFAULT, you may wish to define various keys as different device/directory names.

```
$ DEFINE/KEY/TERMINATE KP1 "WORKDISK:[USER]"  
%DCL-I-DEFKEY, DEFAULT key KP1 has been defined  
$ DEFINE/KEY/TERMINATE KP2 "WORKDISK:[USER.ACCOUNTS]"  
%DCL-I-DEFKEY, DEFAULT key KP2 has been defined
```

Now you can change your default to WORKDISK:[USER.ACCOUNTS] by pressing the PF1 key followed by the keypad number 2 key. The /TERMINATE qualifier of the DEFINE/KEY command places a carriage return after the text; when you press the key, the system attempts to execute the command line. When defining a command line with two or more keys, remember to include all the necessary spaces. In the previous example, the PF1 key definition provides the required space between the command and the parameter.

The informational message following the key definition indicates the key state for which the key is defined (see Section 1.5.3.2). The default key state is DEFAULT. You can suppress the informational message using the /NOLOG qualifier of the DEFINE/KEY command.

### 1.5.3.1 Definable Keys

By default, you can define the top row of keys on the keypad (PF1, PF2, PF3, and PF4). You can also define certain function keys on terminals in the VT200 series. If you set the keypad to application mode (SET TERMINAL/APPLICATION or SET TERMINAL/NUMERIC), you can also define the keypad keys (KP0 through KP9, MINUS, COMMA, PERIOD, and ENTER). See the DEFINE/KEY command in Appendix DCL for a list of terminal keys that you can define.

If you are defining the keypad keys, you may want to define the ENTER key to duplicate the RETURN key.

```
$ DEFINE/KEY/TERMINATE ENTER ""
```

```
%DCL-I-DEFKEY, DEFAULT key ENTER has been defined
```

### 1.5.3.2 Key States

Key states allow a single key to have multiple definitions. Typically, you use a separate key state to define keys with parallel functions. For example, you might define the PF1 key in the DEFAULT state to issue the SET DEFAULT command and change the key state to DIRECTORY. Then, you might define various keys in a second state, DIRECTORY, to issue different directory names. Use the /SET\_STATE qualifier to change the key state temporarily (the key state remains in effect until you press a definable key or terminate the command line). Use the /IF\_STATE qualifier of the DEFINE/KEY command to indicate that a key is defined for the specified state.

```
$ DEFINE/KEY/SET_STATE=DIRECTORY PF1 "SET DEFAULT "
```

```
%DCL-I-DEFKEY, DEFAULT key PF1 has been defined
```

```
$ DEFINE/KEY/TERMINATE/IF_STATE=DIRECTORY MINUS "DISK1:[USER]"
```

```
%DCL-I-DEFKEY, DIRECTORY key MINUS has been defined
```

```
$ DEFINE/KEY/TERMINATE/IF_STATE=DIRECTORY COMMA "DISK1:[ACCOUNTS]"
```

```
%DCL-I-DEFKEY, DIRECTORY key COMMA has been defined
```

To change a key state permanently (until you log out or change the state again), use the /LOCK qualifier with the /SET\_STATE qualifier of the DEFINE/KEY command, or use the /STATE qualifier of the SET KEY command. After permanently changing the key state, you can recall the DEFAULT key state. However, the system does not provide a mechanism that allows you to determine whether the DEFAULT state was the previous key state.

Since you cannot determine the previous key state after permanently changing the key state, you may wish to use the following steps to extend the duration of a temporary state.

1. Use the /SET\_STATE qualifier of the DEFINE/KEY command to change your key state temporarily.
2. Each time you define a key for that temporary state, use the /SET\_STATE qualifier to reset the temporary state.

The following command procedure defines PF3 in the DEFAULT state as the RECALL command and defines each keypad key in the NUMERIC state as the number of the key. The PF3 key, in addition to issuing the RECALL command, sets the key state to NUMERIC temporarily. Since the RECALL command may be followed by any number between 1 and 20, the user must be able to press two NUMERIC keys. To allow a user to press more than one NUMERIC key, each key defined for the NUMERIC state resets the NUMERIC state.

```
$ ! KEYPAD_DEF.COM
$ !
$ IF F$MODE() .NES. "INTERACTIVE" THEN EXIT
$ SET TERM/APPLICATION_KEYPAD
$
$ DEFINE/KEY PF3 "RECALL " /SET_STATE=NUMERIC/NOLOG
$
$ ! Define NUMERIC keypad.
$ DEFINE/KEY KP0 "0" /IF=NUMERIC /SET=NUMERIC /NOLOG
$ DEFINE/KEY KP1 "1" /IF=NUMERIC /SET=NUMERIC /NOLOG
$ DEFINE/KEY KP2 "2" /IF=NUMERIC /SET=NUMERIC /NOLOG
$ DEFINE/KEY KP3 "3" /IF=NUMERIC /SET=NUMERIC /NOLOG
$ DEFINE/KEY KP4 "4" /IF=NUMERIC /SET=NUMERIC /NOLOG
$ DEFINE/KEY KP5 "5" /IF=NUMERIC /SET=NUMERIC /NOLOG
$ DEFINE/KEY KP6 "6" /IF=NUMERIC /SET=NUMERIC /NOLOG
$ DEFINE/KEY KP7 "7" /IF=NUMERIC /SET=NUMERIC /NOLOG
$ DEFINE/KEY KP8 "8" /IF=NUMERIC /SET=NUMERIC /NOLOG
$ DEFINE/KEY KP9 "9" /IF=NUMERIC /SET=NUMERIC /NOLOG
```

### 1.5.3.3 Examining and Deleting Keys

To examine the key definitions that you have created, use the SHOW KEY command. The /DIRECTORY qualifier lists the states that you have defined.

```
$ SHOW KEY/DIRECTORY
DEFAULT
GOLD
```

The /ALL qualifier lists all the keys in the state(s) specified by the /STATE qualifier.

```
$ SHOW KEY/ALL/STATE=DEFAULT
DEFAULT keypad definitions:
  ENTER = "" (noecho,terminate)
  PF1 = "SET DEFAULT " (state=DIRECTORY)
  PF3 = "RECALL " (state=NUMERIC)
```

To delete a key definition, use the DELETE/KEY command. The following command deletes all the keys defined in the GOLD state.

```
$ DELETE/KEY/ALL/STATE=GOLD
```

To delete a particular key, omit the /ALL qualifier and specify the key name as a parameter to the DELETE/KEY command.



### 1.5.4 Recalling Command Lines

As described in Section 1.4.7, CTRL/B allows you to recall the previously typed line. At DCL command level, you can recall the previous 20 command lines. Pressing CTRL/B once recalls the previous command line, pressing CTRL/B again recalls the line before the previous line, and so on to the last saved command line.

To examine up to 20 previously typed command lines, type RECALL/ALL.

```
$ RECALL/ALL
1 SET DEFAULT WORK1:[USER]
2 EDIT ACCOUNTS.COM
3 PURGE ACCOUNTS.COM
4 DIRECTORY/FULL ACCOUNTS.COM
5 COPY ACCOUNTS.COM [ .ACCOUNTS]*
6 SET DEFAULT [ .ACCOUNTS]
```

Having reviewed the available commands, you can recall a particular command line by typing RECALL followed by either the number of the desired command or the first character(s) of the desired command line.

- Number—RECALL returns the command associated with the specified number.  

```
$ RECALL 4
$ DIRECTORY/FULL ACCOUNTS.COM
```
- String—RECALL scans the previous command lines (beginning with the most recent one) and returns the first command line that begins with the specified character string.  

```
$ RECALL DIR
$ DIRECTORY/FULL ACCOUNTS.COM
```

You can also recall a command with the up and down arrow keys, which recall the previous and successive command respectively. For instance, after recalling command number 4 in the preceding example, you can recall command number 3 by pressing the up arrow.

```
$ DIRECTORY/FULL ACCOUNTS.COM
↑
$ PURGE ACCOUNTS.COM
```

Pressing the down arrow after this command recalls command number 4. You can press the arrow keys repeatedly to move rapidly through commands.

## 1.6 User Environment

Each user on the system is associated with a *process*. For example, the system creates a process for you when you log in and deletes that process when you log out. A process contains all the information that the system needs to execute programs, and executes your programs (also called images or executable images) one at a time.

A program executes within the context of the process that invokes it. Some of these programs are system programs that control the flow of events within the process. For example, when you log in, your process is under the control of the system program SYS\$SYSTEM:LOGINOUT.EXE. When you work at DCL command level, your process is under the control of SYS\$SYSTEM:DCL.EXE.

### 1.6.1 Programs

A program is a file of instructions and data in machine-readable format. Image files, which may be supplied by DIGITAL or by users, usually have a file type of EXE. You cannot examine an image file with the DCL commands TYPE, PRINT, or EDIT because image files do not consist of ASCII characters.

A program can be either a command image or a noncommand image.

- **Command image**—A command image is a program that is associated with and invoked by a DCL command. For example, when you type the DCL command COPY, the VAX/VMS system executes the program SYS\$SYSTEM:COPY.EXE; COPY.EXE is a command image. The SYS\$SYSTEM directory contains a number of command image files, most of which are supplied by DIGITAL as part of the VAX/VMS system. (Use the DCL command DIRECTORY SYS\$SYSTEM to examine this directory.)
- **Noncommand image**—A noncommand image is a program that is not associated with a DCL command. To invoke a noncommand image, name the file containing the program as the parameter to the RUN command. The following example invokes the image ACCOUNTS:[INC]INCOME.EXE.

```
$ SET DEFAULT ACCOUNTS:[INC]
$ RUN INCOME
```

Further operations depend on the program. For example, the program might work interactively with you, prompting you for input and displaying information. Or, it might simply run to termination with no outward indication to your terminal. When the program terminates, you are returned to DCL command level.

You can give a noncommand image the appearance of a command image by defining it as a foreign command. Define a foreign command with the following variation of the assignment statement.

```
$ global-symbol-name == "$file-spec"
```

The command name for the foreign command is the global symbol name you specify. The file specification identifies the file containing the noncommand image. The file specification must be prefaced with a dollar sign and include the device, directory, and filename. The file type defaults to EXE. The following example defines INCOME as a foreign command that invokes the program ACCOUNTS:[INC]INCOME.EXE.

```
$ INCOME == "$ACCOUNTS:[INC]INCOME"
```

You can permit the abbreviation of foreign commands by placing an asterisk in the symbol name after the point of the shortest abbreviation. The following example allows you to invoke ACCOUNTS:[INC]INCOME.EXE by typing INC, INCO, INCOM, or INCOME.

```
$ INC*OME == "$ACCOUNTS:[INC]INCOME"
```

A foreign command, since it is a symbol, supersedes any other command with the same name or abbreviation. You cannot invoke a regular command with a name or abbreviation that you have given to a foreign command. You will invoke the foreign command instead.

## 1.6.2 Processes

The environment in which you use the VAX/VMS system is called your process. A process contains identification and status information that the system needs to execute programs for you. Within a process, programs execute one at a time in the order in which they are invoked.

The VAX/VMS system creates a process for you when you:

- Log in—The system creates a process for each interactive user.
- Submit a batch job—The system creates a process for each batch job. When the batch job is completed, the system deletes the process. Section 1.8 discusses batch jobs.
- Spawn a subprocess—The system creates a process when you use the SPAWN command. Section 1.7 describes subprocesses.
- Run a program—The system creates a subprocess when you issue the RUN command with any of the RUN command qualifiers except /DEBUG. The system creates a detached process when you use either the /DETACHED or /UIC=uic qualifiers. See Appendix DCL for more information about the RUN command and its qualifiers.

The system also creates special system processes to perform various functions. The DCL command SHOW SYSTEM displays both user and system processes.

A process can be a detached process (a process that is independent of other processes) or a subprocess (a process that is dependent on another process for its existence and resources).



The following list summarizes the process context (information stored in a process). Use the DCL command `SHOW PROCESS/ALL` to examine your process context.

- Current date and time—The date and time when the `SHOW PROCESS/ALL` command is executed.
- Device name—The name of the device that is allocated to your process. This is usually a terminal.
- User name—The name assigned to the account that is associated with the process.
- Process identification number (PID)—A unique number assigned to the process by the system. The `SHOW PROCESS` command displays the PID as a hexadecimal number.
- Process name—The name assigned to the process. Since process names are unique, the first process logged in under an account is assigned the user name, and subsequent processes logged in under the same account are assigned the terminal name. You can change your process name with the DCL command `SET PROCESS/NAME`.
- User identification code (UIC)—The group and member numbers (and/or letters) assigned to the account that is associated with the process (for example, `[PERSONNEL,RODGERS]` and `[200, 201]`).
- Priority—The current priority of the process.
- Default file specification—The current default device and directory. You can change your current defaults with the DCL command `SET DEFAULT`.
- Devices allocated—The names of the devices allocated to the process. When you are logged in at a terminal, that terminal is allocated to your process.
- Process quotas—The quotas (limits) associated with the process. You can examine these quotas with the `/QUOTAS` or `/ALL` qualifiers of the `SHOW PROCESS` command.
- Accounting information—The continuously updated account of the process's use of memory and CPU time. You can examine this information with the `/ACCOUNTING` or `/ALL` qualifiers of the `SHOW PROCESS` command.
- Process privileges—The privileges associated with the process. You can examine these privileges with the `/PRIVILEGES` or `/ALL` qualifiers of the `SHOW PROCESS` command.
- Process rights identifiers—System-defined identifiers that are used in conjunction with access control list protection (See Section 7.2.2).

- Process dynamic memory area—The process's current use of dynamic memory. You can examine this information with the /MEMORY or /ALL qualifiers of the SHOW PROCESS command.
- Process address space—The virtual memory available to an image executing in your process. (See Section 1.6.3.1.)
- Processes in this tree—A list of subprocesses belonging to the parent process. An asterisk appears after the current process.

### 1.6.3 System Implementation

The following discussion is provided for background information and to explain some of the entries displayed by the DCL commands SHOW MEMORY, SHOW PROCESS, SHOW STATUS, and SHOW SYSTEM. In general, the VAX/VMS system's use of memory is transparent to you.

#### 1.6.3.1 Memory

A program can directly address one gigabyte ( $2^{30}$  bytes) of space for code and data and use most of another gigabyte without too much extra work. This area, together with the space reserved for use by the VAX/VMS operating system, is known as virtual memory. Half of virtual memory, called the process address space, is unique to each process. This is the section of virtual memory that is usually addressed by programs. The other half, called the system address space, is used by the system and shared by all the processes on the system.

Executable images are stored as disk files. When you invoke an image, the VAX/VMS system "maps" the image into your process address space. Mapping associates the virtual addresses named in the image with the physical locations of the code or data at those addresses.

To execute the image, the VAX/VMS system must bring the code and data into physical memory. Typically, the system does not have enough physical memory to hold every process's image, plus the system code and data structures. Therefore, the VAX/VMS system restricts each process to a certain number of pages in physical memory (a page is 512 bytes). This set of pages is called the process's working set. The collection of working sets currently in memory is called the system balance set.

The DCL command SHOW STATUS displays the size (in pages) of your process working set. The DCL command SHOW MEMORY displays the number of processes that are resident in the balance set. The following table lists the main components of physical memory in a VAX/VMS system.

Resident System	Page Cache	Balance Set
Executive routines	Free-page list	Process working sets
Nonpaged dynamic memory	Modified-page list	System working sets
PFN database		Pageable exec
System Page Table		System message file
		Paged dynamic memory
		Record Management Services

### 1.6.3.2 Images and Working Sets

Once an image has been mapped into process address space, it can be executed. Code or data is accessed from the pages in the working set as it is needed by the image. If the code or data is not on a page in the working set, a page fault occurs, causing the VAX/VMS system to read pages into the working set from a disk file. Since a working set is empty when image execution begins, the first statements in an image generate a lot of page faults. Usually, as the working set fills up, page faults are generated at a slower rate. The DCL command SHOW SYSTEM displays the total number of page faults generated by each process on the system. If a page fault occurs when your working set is full, the VAX/VMS system must remove some pages in the working set to make room for new pages. When the system removes a page from your working set, the page is placed in one of the page caches. If the page was modified while it was in the working set, it is placed on the modified-page list; if it was not modified, it is placed on the free-page list.

When the modified-page list fills up, the system copies the contents of pages in the modified-page list to the paging file on disk. Then, the pages on the modified-page list are moved to the free-page list. A page on the free-page list is available to any process. For example, if a process needs a page of physical memory, a page on the free-page list can be overwritten (the contents replaced by new code or data). Or, if a process needs the contents of a page on the free-page list, the page can be faulted back into the working set of the appropriate process.

Pages can be faulted into the working set from the disk or from the page cache if the page had been faulted out and was not reused. Since the page cache is in physical memory, restoring pages from the page cache is faster than reading pages from the disk. The DCL command SHOW MEMORY displays the number of pages in the page cache.

### 1.6.3.3 Processes and the Balance Set

If enough processes are on the system, the system may run out of physical memory and/or balance set slots. If either of these events occurs, the VAX/VMS system does one of the following:

- **Trims working sets**—The system adjusts the size of one or more working sets. Usually, pages are taken from a number of different working sets and placed in the page file to make room in physical memory.
- **Swaps processes**—One or more processes are swapped out (removed from the balance set). When a process is swapped out, its working set is placed in the swapping file. A process cannot execute while it is swapped out.

The VAX/VMS system examines a process's scheduling state and priority to decide whether a process should be swapped into or out of the balance set. In general, processes ready to execute are kept in the balance set while processes in one of the wait states are more likely to be swapped out. The DCL command `SHOW SYSTEM` displays the state and priority of each process on the system. The following table lists the possible process-scheduling states.

State	Explanation
CEF	Common event flag wait. Process is waiting for a combination of common event flags to be set.
COLPG	Collided page wait. Page fault occurred while the requested page was being read into physical memory for another process; process is waiting for the page to be read in.
COM	Computable. Process is ready to execute.
COMO	Computable, outswapped. Process is ready to execute but swapped out.
CUR	Current. Process is currently executing.
FPG	Free-page wait. Process is waiting for a free page of physical memory.
HIB	Hibernate wait. Process is hibernating.
HIBO	Hibernate wait, outswapped. Process is hibernating and swapped out.
LEF	Local event flag wait. Process is waiting for a combination of local event flags (usually, I/O related) to be set.
LEFO	Local event flag wait, outswapped. Process is waiting for a combination of local event flags (usually I/O related) to be set and swapped out.
MUTEX	Mutual exclusion semaphore wait. Waiting to ensure that only one process has access to requested code.
PFW	Page fault wait. Page fault has occurred; process is waiting for the page to be read into physical memory.
SUSP	Suspend wait. Process is suspended.

State	Explanation
SUSPO	Suspend wait, outswapped. Process is suspended and swapped out.
RWxxx	Resource wait. The characters following the RW indicate the source of the wait: <ul style="list-style-type: none"> <li>BRK (broadcast)</li> <li>CLU (cluster translation)</li> <li>IMG (image activation)</li> <li>LCK (lock ID database)</li> <li>MBX (mailbox)</li> <li>MPB (modified-page writer)</li> <li>MPE (modified-page list)</li> <li>NPG (nonpaged pool)</li> <li>PAG (paged pool)</li> <li>PFF (page file)</li> <li>QUO (pooled quota)</li> <li>SWP (swap file)</li> </ul>

#### 1.6.3.4 Resident System

The resident system is always in physical memory. It consists of:

- Executive routines—Routines that must be available to the VAX/VMS system at all times
- Nonpaged dynamic memory—System data structures and code that are shared among all the processes on the system

#### 1.6.3.5 System Parameters

When your VAX/VMS system is first installed, the system adjusts certain system parameters, called SYSGEN parameters, for your system hardware. You can alter these parameters by setting values with the System Generation Utility (SYSGEN).

## 1.7 Subprocesses

To create a subprocess, use the DCL command SPAWN. By default, the subprocess is created and you are given control at DCL command level within the subprocess. Your default directory is the current default directory of the parent process. (You can suppress the informational messages from DCL by specifying the /NOLOG qualifier of the SPAWN command.)

```
$ SPAWN
%DCL-S-SPAWNED, process USER_1 spawned
%DCL-S-ATTACHED, terminal now attached to process USER_1
$
```



Alternatively, you can use SPAWN's command parameter and /INPUT qualifier to define SYS\$COMMAND (command stream) and SYS\$INPUT (data stream) for the subprocess. SYS\$COMMAND and SYS\$INPUT determine subprocess execution.

- No parameter, no /INPUT—SYS\$COMMAND and SYS\$INPUT default to your terminal, as shown in the previous example. (SYS\$INPUT defaults to NL: if you invoke SPAWN from a command procedure.)
- Parameter, but no /INPUT—SYS\$COMMAND is the command parameter (either a command or a command procedure) and SYS\$INPUT defaults to your terminal. (SYS\$INPUT defaults to NL: if you invoke SPAWN from a command procedure.) If the command parameter is a command that requires data, data is read from the terminal. If the command parameter is a command procedure that requires data, how data is read depends on how the command procedure is written (INQUIRE, READ, and so on).
- /INPUT, but no parameter—SYS\$COMMAND and SYS\$INPUT are both defined as the file or device you specify as the qualifier value. If the specified command(s) require data, the data is read from the specified file or device. In addition, when you use the /INPUT qualifier of the SPAWN command to specify a nonterminal input stream, the subprocess is created as a noninteractive process. A noninteractive process exits upon encountering a severe error or an end-of-file indicator. At DCL command level, CTRL/Z is treated as an end-of-file indicator.
- Parameter and /INPUT—SYS\$COMMAND is the command parameter (either a command or command procedure) and SYS\$INPUT is the file or device you specify as the /INPUT qualifier value. If a command specified by the command parameter requires data, data is read from SYS\$INPUT.

The following command creates a subprocess to execute the command procedure SORT.COM. If SORT requires any data, data is read from the terminal. When the command procedure completes, the subprocess is deleted along with any subprocesses that it may have created.

```
$ SPAWN @SORT.COM
%DCL-S-SPAWNED, process USER_1 spawned
%DCL-S-ATTACHED, terminal now attached to process USER_1

. ! output from SORT.COM
.

%DCL-S-RETURNED, control returned to process USER
```

The following command creates a subprocess to execute the command procedure SORT.COM. If SORT requires any data, the data must be supplied by the command procedure (as in a batch job). When the command procedure completes (or encounters a severe error), the subprocess is deleted along with any subprocesses that it may have created.

```
$ SPAWN/INPUT=SORT
```



Because each process that you create is unique, commands executed in one process do not usually affect any other process. However, since control of the terminal passes between processes, commands that affect the terminal characteristics (SET TERMINAL) affect any process controlling that terminal. For example, if one process inhibits echoing and exits without restoring it, echoing remains inhibited for the next process that gains control of the terminal.

### 1.7.1 Exiting from a Subprocess

To exit from a subprocess created by SPAWN, use one of the following commands:

- **LOGOUT**—When you exit from a subprocess with the LOGOUT command, the subprocess is deleted (along with any subprocesses that it created) and you are returned to the parent process.
- **ATTACH**—When you exit from a subprocess with the ATTACH command, the subprocess hibernates and control of your terminal is transferred to the specified process. (You must either specify a process name as a parameter to the ATTACH command or a process identification number as a value of the /IDENTIFIER qualifier of the ATTACH command.) The following example shows how to exit from the subprocess USER\_1 and attach to the process USER using the /IDENTIFIER qualifier.

**\$ SHOW PROCESS**

```
26-APR-1986 10:37:27.32                User: USER
Pid: 00000019  Proc. name: USER_1      UIC: [200,200]
Priority: 4    Default file spec: SYS$SYSDEVICE:[USER]
```

**\$ ATTACH/IDENTIFIER=00000018**

%DCL-S-RETURNED, control returned to process USER

**\$ SHOW PROCESS**

```
26-APR-1986 10:34:58.50  OPA0:                User: USER
Pid: 00000018  Proc. name: USER          UIC: [200,200]
Priority: 4    Default file spec: SYS$SYSDEVICE:[USER]
```

Devices allocated: OPA0:

### 1.7.2 Subprocess Context

A subprocess's context is copied from the parent's context with the following exceptions:

- **Process identification number**—The system assigns each created subprocess a unique process identification number.
- **Process name**—By default, the subprocess name consists of the name of the parent process followed by an underscore and an integer. Use the `/PROCESS` qualifier of the `SPAWN` command to specify a process name other than the default. A process name must be unique.
- **Created commands**—Commands that are defined by a parent process using the `SET COMMAND` command are not copied to a subprocess. To use a created command in a subprocess, you must use `SET COMMAND` to create that command for the subprocess.

The previous list implies that, by default, a created subprocess inherits the following items from the parent process: defaults, privileges, symbols, logical names, control characters, message format, verification state, and key definitions. You can use the following `SPAWN` qualifiers to prevent the subprocess from inheriting a number of these items.

Qualifier	Items Inhibited or Changed
<code>/CARRIAGE_CONTROL</code> , <code>/PROMPT</code>	DCL prompt
<code>/NOCLI</code>	CLI (DCL by default)
<code>/NOKEYPAD</code>	Keypad definitions
<code>/NOLOGICAL_NAMES</code>	Logical names
<code>/NOSYMBOL</code>	Symbols

**NOTE:** The `/SYMBOL` and `/LOGICAL_NAMES` qualifiers do not affect system-defined symbols (such as `$SEVERITY` and `$STATUS`) or system-defined logical names (such as `SYS$COMMAND` and `SYS$OUTPUT`).

Since copying logical names and symbols to a subprocess can be time consuming (a few seconds), you will probably want to use the `/NOLOGICAL_NAMES` and `/NOSYMBOL` qualifiers unless you plan to use the logical names or symbols in the subprocess. If you create subprocesses frequently, the best practice is to use the `ATTACH` command to transfer control between the parent and the subprocess rather than repeatedly waiting for the system to create a new subprocess for you.

### 1.7.3 Subprocess Execution

Typically, you use a subprocess in one of two ways:

- **Interrupts**—To perform a second task, then return to an original task. Since SPAWN is a built-in command (see Section 1.4), you can use CTRL/Y to interrupt one task, spawn a subprocess to perform a second task, exit from the subprocess, and enter the CONTINUE command to return to the original task. Note that you can only issue the CONTINUE command after interrupting a command procedure when there is no ON\_CONTROL\_Y condition in the command procedure. (If you interrupt the EDT editor, enter the CONTINUE command and then press CTRL/W to refresh the screen.)
- **Batch**—To perform a second task while continuing to work on your original task.

By default, when you create a subprocess, the parent process hibernates. When you return to the parent process, you either delete or hibernate the subprocess. Only one of your processes is executing at any one time.

To execute a subprocess and its parent process at the same time, you must specify the /NOWAIT qualifier of the SPAWN command. Since both processes are executing concurrently, both will attempt to control the terminal. To prevent conflicts, when you create a subprocess using the /NOWAIT qualifier, also specify the following:

- **/OUTPUT qualifier**—Indicates that the subprocess should write output to a specified file rather than to the terminal.
- **SPAWN command parameter or /INPUT qualifier**—Indicates that the subprocess should execute the specified commands rather than reading input from the terminal.

When you specify the /INPUT qualifier of the SPAWN command, the subprocess is created as a noninteractive process that exits upon encountering a severe error or an end-of-file indicator (such as CTRL/Z pressed at DCL command level).

## 1.8 Batch Jobs

A batch job consists of one or more command procedures that execute in a detached process without user interaction. A batch queue is a list of batch jobs waiting to execute. Your job may have to wait for a previous job to complete before it can begin executing. (The number of batch jobs that can execute simultaneously is specified when the batch queue is created by the system manager.)

In general, you use batch jobs to execute a command or set of commands that either take a long time to execute or that you want to schedule for execution after regular hours (see Section 1.8.2).

### 1.8.1 Submitting a Batch Job

Use the SUBMIT command to enter a command procedure into a batch queue. If you specify more than one command procedure, the procedures execute one after another in the order of specification. The following command enters LIBRARY.COM and SORT.COM into the default batch queue SYS\$BATCH (file type defaults to COM).

```
$ SUBMIT LIBRARY, SORT
```

```
Job LIBRARY (queue SYS$BATCH, entry 201) started on SYS$BATCH
```

The system displays the name of the job, the queue containing the job, and the entry number assigned to the job. Once a job is submitted, you reference it using the entry number.

To specify parameters for a command procedure submitted as a batch job, use the /PARAMETERS qualifier of the SUBMIT command. Note that you can also pass data to a batch job by including the data in the command procedure itself or by defining SYS\$INPUT to be a file. (See Section 6.3 for further information.) The specified parameters are used for each command procedure in the batch job. You cannot specify different parameters for individual command procedures within a single job. The following example passes LIBRARY.COM and SORT.COM the parameters WORKDISK:[ACCOUNT.BILLS]DATA.DAT and WORKDISK:[ACCOUNT]NAME.DAT.

```
$ SUBMIT LIBRARY, SORT/PARAMETERS= -
```

```
_$ (WORKDISK: [ACCOUNT.BILLS]DATA.DAT, WORKDISK: [ACCOUNT]NAME.DAT)
```

```
Job SORT (queue SYS$BATCH, entry 203) started
```

Your batch job executes as if you had logged in and executed each of the submitted command procedures. For example, the previous SUBMIT command executes a batch job that logs in under your account, executes your login command procedure, and then executes the following commands:

```
$ @LIBRARY WORKDISK: [ACCOUNT.BILLS]DATA.DAT WORKDISK: [ACCOUNT]NAME.DAT
```

```
$ @SORT WORKDISK: [ACCOUNT.BILLS]DATA.DAT WORKDISK: [ACCOUNT]NAME.DAT
```

Since your login defaults are not usually the defaults needed to access files mentioned in a command procedure, the command procedures should use one of the following mechanisms to ensure that the correct files are accessed.

- Use complete file specifications—When accessing a file in a command procedure or passing a file to a command procedure, include the device and directory names as part of the file specification, as shown in the previous example.
- Use the SET DEFAULT command—Before accessing a file in a command procedure, use the SET DEFAULT command to specify the proper device and directory.



```
$ ! SORT.COM
$ !
$ ! Sort P1 into P2
$ SET DEFAULT WORKDISK: [ACCOUNT.BILLS]
$ SORT/KEY=(POSITION:1,SIZE:4) 'P1' 'P2'
```

### 1.8.2 Controlling a Batch Job

To specify control information when submitting a batch job, add the appropriate qualifiers to the SUBMIT command. To specify control information after submitting a batch job but before the job begins executing, use the SET QUEUE/ENTRY=n command (where n is the job number). Control information allows you to perform a number of operations; the following list describes the most common operations. (Unless otherwise noted, the specified qualifiers can be used with either SUBMIT or SET QUEUE/ENTRY. For a complete list of qualifiers see the command descriptions in Appendix DCL.)

- Specify job name—Use the /NAME qualifier. The job name defaults to the file name of the first command procedure.
- Delay execution—To specify an exact time of execution, use the /AFTER qualifier. To hold a job indefinitely, use the /HOLD qualifier. To execute a job that is being held (either by /AFTER or /HOLD), use the /RELEASE or /NOHOLD qualifier of the SET QUEUE/ENTRY command.
- Delete job—Use the DELETE/ENTRY command if the job has not started. Use the STOP/QUEUE/ENTRY command if the job is executing.
- Restart job—Use the STOP/QUEUE/REQUEUE/ENTRY command. You can restart the job on the same queue or on a different queue. A batch job cannot be restarted unless it was submitted with the /RESTART qualifier. See Section 1.8.4 for more information on restarting batch jobs.
- Delete command procedures—Use the /DELETE qualifier. As a command qualifier, /DELETE indicates that all command procedures are to be deleted after executing. As a file qualifier, /DELETE indicates the specified file is to be deleted after executing.
- Request notification of job completion—Use the /NOTIFY qualifier. When the job completes, a message to that effect is broadcast to the terminal that you are logged in on.

Note that if multiple procedures are submitted in a batch job, the batch job terminates when any procedure exits with an error or fatal error status.

### 1.8.3 Batch Job Output

By default, accumulated output from a batch job is written to a log file once each minute. (To specify a different time interval, include the `SET OUTPUT_RATE` command in your command procedure.) If you attempt to read the file while the system is writing to it, you receive a message indicating that the file is locked by another user. Wait a few seconds and try again.

The output from a batch job includes the contents of your login command file, everything written to `SYS$OUTPUT` (command procedure output, error messages, and so on), and the full logout message. To prevent your login command procedure from being written to the batch log file, begin your login command procedure with the following command:

```
$ IF F$MODE() .EQS. "BATCH" THEN SET NOVERIFY
```

The name of the log file defaults to the job name, a file type of `LOG`, and the device and directory specified by your login defaults. To specify a different log file name when you submit the job, use the `/LOG_NAME` qualifier of the `SUBMIT` command.

When the batch job completes, the log file is queued to `SYS$PRINT`, printed, and deleted. To save the log file after printing it, use the `/KEEP` qualifier of the `SUBMIT` command. To save the log file without printing it, use the `/NOPRINT` qualifier of the `SUBMIT` command.

### 1.8.4 Restarting Batch Jobs

By default, if the system fails while your batch job is executing, your job does not complete. When the system recovers and the queue is restarted, your job is aborted and the next job in the queue is executed. However, by specifying the `/RESTART` qualifier when you submit a job (`SUBMIT` command), you indicate that the system should reexecute the job if the system crashes before the job completes. In addition, specifying the `/RESTART` qualifier allows you to abort execution of the job and restart it on the same queue or a different queue using the `/ENTRY` qualifier of the `STOP/QUEUE/REQUEUE` command.

By default, a batch job is reexecuted beginning with the first line. You can use the following symbols to specify a different restarting point:

- `$RESTART`—A global symbol whose value is true if the batch job has been started at least once before this execution. Do not specify a value for `$RESTART`; the system will assign the appropriate value.
- `BATCH$RESTART`—A global symbol whose value you specify with the `SET RESTART_VALUE` command.



The following steps describe how to use these symbols in a command procedure.

- Begin each possible starting point of the procedure with a label.
- As the first step in each section, equate the value of BATCH\$RESTART to the label with the SET RESTART\_VALUE command.
- At the beginning of the procedure, test \$RESTART. If \$RESTART is true, issue a GOTO statement using BATCH\$RESTART as the transfer label.

The following command procedure extracts a number of modules from a library, concatenates those modules, and then sorts the resulting file. If aborted, the command procedure reexecutes from the beginning of the file, the statement labeled CONCATENATE\_LIBRARIES or the statement labeled SORT\_FILE, depending on the value of BATCH\$RESTART. (If you were extracting a number of separate modules, you could make each extraction a separate section.)

```
$ ! SORT_MODULES.COM
$ !
$ ! set default to the directory containing
$ ! the library whose modules are to be sorted
$ SET DEFAULT WORKDISK: [ACCOUNTS.DATA83]
$
$ ! check for restarting
$ IF $RESTART THEN GOTO BATCH$RESTART
$
$ EXTRACT_LIBRARIES:
$ SET RESTART_VALUE=EXTRACT_LIBRARIES
.
.
.
$ CONCATENATE_LIBRARIES:
$ SET RESTART_VALUE=CONCATENATE_LIBRARIES
.
.
.
$ SORT_FILE:
$ SET RESTART_VALUE=SORT_FILE
.
.
.
$ EXIT
```

## 1.9 Error Conditions

System commands and utilities place a value in the global symbol `$STATUS` when they exit. User programs may also set `$STATUS`. To set `$STATUS` when exiting from your command procedure, include a parameter value on the `EXIT` command. The system uses `$STATUS` to determine which message, if any, to display and whether or not to continue execution. The value of the lower three bits in `$STATUS` is placed in the global symbol `$SEVERITY`. This value indicates success or failure, as shown:

Value	Severity
0	Warning
1	Success
2	Error
3	Information
4	Fatal or severe error

In the following example, `$STATUS` and `$SEVERITY` indicate successful completion of a command.

```
$ SHOW SYMBOL $STATUS
$STATUS = "%X00000001"
$ SHOW SYMBOL $SEVERITY
$SEVERITY = "1"
```

Success is indicated by an odd value; failure and warning are indicated by an even value. Therefore, `$SEVERITY` (or `$STATUS`) can be used as a logical value. The following statement (from a command procedure) passes control to an error routine if `$SEVERITY` indicates failure.

```
$ IF .NOT. $SEVERITY THEN GOTO ERROR
```

### 1.9.1 Commands That Do Not Change `$STATUS`

The following commands do not change `$STATUS` or `$SEVERITY` when they execute successfully:

CONTINUE	DECK	DEPOSIT	EOD
EXAMINE	GOTO	IF	SHOW STATUS
SHOW SYMBOL	STOP	WAIT	

## 1.9.2 System Error Messages

System error messages are displayed in the format:

%FACILITY-L-IDENT, text

FACILITY = the mnemonic for the program issuing the message

L = the first letter of the severity code

IDENT = an abbreviation of the text

text = an explanation of the error

Suppress any part of the error message with the SET MESSAGE command. A SET MESSAGE command remains in effect until you enter the SET MESSAGE command again or log out. The following command suppresses the explanatory text of the message.

```
$ SET MESSAGE/NOTEXT
```

To examine the current setting of SET MESSAGE, use the MESSAGE keyword of the F\$ENVIRONMENT lexical function.

```
$ MESSAGE = F$ENVIRONMENT ("MESSAGE")
```

```
$ SHOW SYMBOL MESSAGE
```

```
MESSAGE = "/FACILITY/SEVERITY/IDENTIFICATION/NOTEXT"
```

## 1.10 Using the Mail Utility

The interactive Mail Utility (MAIL) allows you to send and receive messages, as well as to file, forward, delete, and reply to messages that you have received. To invoke the interactive Mail Utility, specify the DCL command MAIL without parameters.

```
$ MAIL
```

```
MAIL>
```

You can display information about MAIL commands by entering HELP in response to the MAIL> prompt (or you can refer to Appendix MAIL for descriptions of MAIL commands). To exit from MAIL, enter the MAIL command EXIT or press CTRL/Z.

### 1.10.1 Sending Mail

You can create and send mail messages interactively with the Mail Utility. You can send files to other users from within the Mail Utility or from the DCL command level.

### 1.10.1.1 Sending a Mail Message

After invoking the Mail Utility, specify the SEND command to create and send a mail message. MAIL prompts you for the names of the users to whom you want to send the message, the subject of the message (optional), and the text of the message (optional). The following example sends a message to user name USER:

```
MAIL> SEND
To:      USER
Subj:    Meeting of June 9
Enter your message below.Press CTRL/Z when complete,or CTRL/C to quit:
Sorry I cannot make the meeting: I'll be on vacation
during that week. Let me know how it goes.
                        Frank
```

Note that pressing CTRL/Z actually sends the message.

If you decide not to send the message, enter CTRL/C, which cancels the SEND operation without exiting from MAIL.

To use the editor when sending a mail message interactively, specify the /EDIT qualifier with the SEND command.

```
MAIL> SEND/EDIT
```

After you respond to the prompts for the names to whom you want to send the message and for the subject of the message, MAIL invokes the editor, which you can use to compose your message. By default, MAIL invokes the EDT editor. Section 1.10.1.3 describes how to change the default editor. To send the message, exit from the editor by pressing CTRL/Z and entering the EXIT command; to cancel the send operation, exit from the editor by pressing CTRL/Z and entering the QUIT command.

In addition to the SEND command, you can use the /EDIT qualifier with the REPLY and FORWARD commands. By specifying /EDIT when you invoke the Mail Utility, you can use the editor for SEND, REPLY, and/or FORWARD operations during the ensuing mail session. See Appendix MAIL for descriptions of the /EDIT qualifier for the Mail Utility commands.

### 1.10.1.2 Sending a File

To send a file to one or more users, specify either the DCL command MAIL or the SEND command of the Mail Utility. The following DCL command sends the file MEMO.TXT to user name USER. The argument to the /SUBJECT qualifier must be enclosed in quotation marks if it contains any spaces or nonalphanumeric characters.

```
$ MAIL/SUBJECT="Another memo" MEMO.TXT USER
```

If you use the SEND command from within interactive MAIL, you can specify the /EDIT qualifier to edit the file before sending it. The following example invokes the editor to edit the file MEMO.TXT before sending the file to user name USER.

```
MAIL> SEND/EDIT MEMO.TXT
To:      USER
Subj:    The memo I promised you.
```

To send the message, press CTRL/Z and enter the EXIT command; to cancel the SEND operation, press CTRL/Z and enter the QUIT command.

### 1.10.1.3 Setting the Default Editor

By default, MAIL invokes the EDT editor whenever you specify the /EDIT qualifier. To specify an editor other than EDT as the default editor for MAIL, do the following:

1. Copy the MAILEDIT.COM command procedure to your directory with the following command:  

```
$ COPY SYS$SYSTEM:MAILEDIT.COM *
```
2. Edit the command procedure MAILEDIT.COM, changing each occurrence of the EDIT command to the command name you have defined to invoke your alternative editor. For example, if you use the foreign command WINK to invoke your editor, the edited version of MAILEDIT.COM should read as follows (see Section 1.6.1 for information on defining foreign commands).

```
$ ! MAILEDIT.COM
$ !
$ ! Command procedure to invoke an editor for MAIL.
$ !
$ ! Inputs:
$ !
$ !      P1 = Input filename.
$ !      P2 = Output filename.
$ !
$ ! If MAIL$EDIT is undefined, MAIL will invoke
$ !      callable EDT.
$ ! If MAIL$EDIT is defined to be a command procedure,
$ ! MAIL will create a subprocess to edit the mail.
$ !
```



## 1-52 Interaction with the System

```
$ ! Note that this procedure is run in the context
$ !   of a subprocess.
$ ! LOGIN.COM is not executed. However, all process
$ !   logical names and DCL global symbols are
$ !   copied.
$ !
$ ! The default directory is the same as that
$ !   of the parent process.
$ !
$ DEFINE /USER SYS$INPUT 'F$TRNLNM("SYS$OUTPUT")'
$ IF P1 .EQS. "" THEN GOTO NOINPUT
$ WINK /OUTPUT='P2' 'P1'
$ EXIT
$NOINPUT:
$ WINK 'P2'
$ EXIT
```

3. Include the following line in your LOGIN.COM file.

```
$ DEFINE MAIL$EDIT disk:[directory]MAILEDIT.COM
```

### 1.10.1.4 Sending a Message Between Systems

To send a message to a user on another system, specify the node name of that system, a double colon, and the user name.

nodename::username

For example, to send a message to a user named OSGOOD on a node named BRUTUS, specify:

```
MAIL> SEND
To:      BRUTUS::OSGOOD
```

### 1.10.1.5 Sending a Message to a Distribution List

You can create a file containing a list of users to whom you frequently send mail and specify the file name rather than the individual user names when you send mail to those users. The default file type of a distribution list file is DIS.

When you create a distribution list, type one user name per line. You can also include the names of other distribution lists by specifying an at sign (@) followed by the name of the distribution list. Begin comments with exclamation points. For example:

```

! ALLBUD.DIS
!
! Budget Committee Members
@BUDGET ! listed in BUDGET.DIS
! Staff
USER ! me
BRUTUS::WILSON ! Martha Wilson
PORTIA::RIPLEY ! Roy Ripley

```

To use the distribution list, type an at sign (@) and then the file name in response to the To: prompt. For example:

```

MAIL> SEND
To:      @ALLBUD
Subj:    Tomorrow's Meeting

```

You can also specify more than one distribution list in response to the To: prompt.

```

MAIL> SEND
To:      USER,@ALLBUD
Subj:    Tomorrow's Meeting

```

## 1.10.2 Reading Mail

You must invoke the interactive Mail Utility to read a mail message. Messages that you receive are stored in mail files, which have a default file type of MAI. Your default mail file, MAIL.MAI, is created in your default directory the first time you receive a mail message. See Section 1.10.5.3 for information on creating, accessing and deleting mail files.

### 1.10.2.1 New Messages

When you are logged in and receive a mail message, notice of the new message appears on your screen. For example, a message sent by user name USER would appear as:

```
New mail from USER
```

You are also notified that you have new mail when you log in and when you invoke MAIL. To read a new mail message, invoke MAIL interactively; MAIL prompts for a command and, if you have received mail, displays the number of mail messages you have received.

```
$ MAIL
```

```
You have 1 new message.
```

```
MAIL>
```

## 1-54 Interaction with the System

To read the new message, press RETURN; the message appears on your screen.

```
#1                1-JUN-1986  14:12:27          NEWMAIL
```

```
From:  USER
To:    BOSS
Subj:  Meeting of June 9
```

The meeting has been moved from the auditorium to the cafeteria.

MAIL>

To continue reading your new mail messages, press RETURN in response to the MAIL> prompt. Pressing RETURN in MAIL is equivalent to specifying the READ command without parameters. When you have read all new messages, MAIL issues the message "%MAIL-E-NOMOREMSG, no more messages," and continues to prompt for commands until you exit by entering EXIT or pressing CTRL/Z.

If you receive a mail message while you are in interactive MAIL, specify the READ /NEW command to read the new message.

### 1.10.2.2 Old Messages

To read old mail messages, invoke the Mail Utility and press RETURN. If you have no new mail messages, the first message (numbered 1) in your default mail file appears on your screen. Press RETURN to display the next message. If the message is too long to display on the screen, pressing RETURN displays the next part of the message. To skip part of a message and display the next message, specify the NEXT command.

You can select and display a particular message within the current folder by specifying the READ command and the number of the message. (Section 1.10.5 describes folders.) To list the numbers of messages in the current mail folder, specify the DIRECTORY command.

MAIL> DIRECTORY

#	From	Date	Subject	MAIL
1	BRUTUS::OSGOOD	17-NOV-1986	our meeting	
2	USER	17-NOV-1986	status	

MAIL> READ 2

If you have a large number of messages, you can locate a particular message by using the SEARCH command to find a specified string. The SEARCH command selects and displays the first message in the current folder that contains the specified string. To search for a new string, specify the string as a parameter of the SEARCH command. Each time you specify a new string, the SEARCH command starts the search at message number 1. To continue searching the folder for messages that contain the specified string, use the SEARCH command without specifying a parameter.

MAIL> **SEARCH STAFF**

Selects and displays the first message containing "staff."

MAIL> **SEARCH**

Selects and displays the next message containing "staff."

MAIL> **SEARCH**

%MAIL-E-NOTFOUND, no messages containing 'STAFF' found

### 1.10.3 Creating a File from a Mail Message

To copy a mail message to a sequential file, use the EXTRACT command. The following commands create a file named MEETING.TXT containing the text of message number 3. (The /NOHEADER qualifier specifies that only the text of the message is to be copied to the file.) In this case, BOSS is a logical name you have previously defined (using the DEFINE command) to equate to your supervisor.

MAIL> **READ 3**

#3                                      1-JUN-1986   14:12:27                      NEWMAIL

From:    USER

To:       BOSS

Subj:    Meeting of June 9

The meeting has been moved from the auditorium to the cafeteria.

MAIL> **EXTRACT/NOHEADER MEETING.TXT**

Use the /APPEND qualifier of the EXTRACT command to copy a message to the end of an existing file. Use the /ALL qualifier to copy all the files in the current folder to an existing file (Section 1.10.5 discusses folders).

### 1.10.4 Deleting Mail

To delete a mail message, either specify the number of the mail message as the parameter of the DELETE command, or read the message (to make it the current message) and specify the DELETE command without parameters. For example, to select and delete message number 3, specify:

MAIL> READ 3

#3                      1-JUN-1986   14:12:27                      NEWMAIL

From: USER  
To: BOSS  
Subj: Meeting of June 9

The meeting has been moved from the auditorium to the cafeteria.

MAIL> DELETE

The following command has the same effect as the preceding example.

MAIL> DELETE 1

When you delete a message, the message is moved to the wastebasket folder. During your interactive MAIL session you can recover any deleted message by moving the message out of the wastebasket folder (see Section 1.10.5.1). When you exit from the current mail file (either by exiting from MAIL or by specifying a different mail file), MAIL automatically deletes all messages in the wastebasket folder. (Section 1.10.5.3 discusses mail files.)

By default, the wastebasket folder is named WASTEBASKET. Use the SET WASTEBASKET\_NAME command to specify a different name for the wastebasket folder. Use the SHOW WASTEBASKET\_NAME command to display the current name of the wastebasket folder.

## 1.10.5 Organizing Mail

By default, each account has one mail file (MAIL.MAI in your login default directory) that contains two folders:

- NEWMAIL—Contains all messages that have not been read. If you enter mail and NEWMAIL contains one or more messages, your current folder will be NEWMAIL. Once you leave NEWMAIL (either by exiting from MAIL or by changing folders), MAIL moves any messages that have been read and not deleted to your MAIL folder; the messages moved to MAIL are deleted from NEWMAIL.
- MAIL—Contains messages that have been read and not deleted. If you invoke mail and NEWMAIL contains no messages, your current folder will be MAIL.

You can create any number of mail files. Each mail file can contain any number of folders. Each folder can contain any number of messages. Typically, you organize your messages by creating folders rather than by creating mail files. If your mail file is very large (over 500 blocks), you may want to create separate mail files for the larger folders to improve the Mail Utility's performance.



### 1.10.5.1 Creating and Modifying Folders

The following commands allow you to create and modify folders.

- **COPY**—Places a copy of the current message into the specified folder. If the folder does not exist, you are prompted for confirmation before it is created. The following commands copy all the messages concerning the FEED account from the current folder to a folder named FEED. After the commands are executed, you have two copies of each message, one in the current folder and one in FEED.

```
MAIL> SEARCH FEED
```

Selects and displays first message containing "feed."

```
MAIL> COPY FEED
```

Folder FEED does not exist.

Do you want to create it (Y/N, default is N)?Y

```
%MAIL-I-NEWFOLDER, folder FEED created
```

```
MAIL> SEARCH
```

Selects and displays next message containing "feed."

```
MAIL> COPY FEED
```

```
MAIL> SEARCH
```

```
%MAIL-E-NOTFOUND, no messages containing 'FEED' found
```

- **MOVE**—Moves the current message into the specified folder. If the folder does not exist, you are prompted for confirmation of its creation. After being moved, the message is automatically deleted from the current folder. The following command moves the current message to the folder named FEED, deleting it from the current folder.

```
MAIL> MOVE FEED
```

- **FILE**—Interchangeable with the MOVE command.

### 1.10.5.2 Deleting Folders

To delete a mail folder, delete all the messages in the folder. The following example deletes the FEED folder.

```
MAIL> SELECT FEED
```

```
%MAIL-I-SELECTED, 2 messages selected
```

```
MAIL> DELETE/ALL
```

### 1.10.5.3 Creating, Accessing, and Deleting Mail Files

To create a mail file, move a message into the file. Use the COPY, MOVE, or FILE command as you would to create a folder with the following addition: specify the name of the mail file after the name of the folder. The following example creates the mail file ACCOUNTS.MAI and moves the current message into a folder named FEED in the file ACCOUNTS.MAI, deleting the message from its current folder and file.

```
MAIL> MOVE FEED ACCOUNTS
```

To work within a mail file other than the default mail file, use the SET FILE command to specify the alternate file. (The SHOW FILE command displays the name of the current mail file.) When you change mail files, the WASTEBASKET folder of the current mail file is emptied and the mail file is closed.

To delete a mail file, delete all the messages in all the folders in the mail file (or copy, file, or move them to another mail file). Then exit from MAIL, change the protection of the file and delete it (using the DCL commands SET PROTECTION and DELETE).

### 1.10.5.4 Selecting Folders

The name of the current folder is displayed in the top right corner of the screen each time you enter a READ or DIRECTORY command. You can operate only on messages that are in your current folder.

To display a list of the folders in your current mail file, use the DIRECTORY /FOLDER command.

```
MAIL> DIRECTORY/FOLDER
```

```
Listing of folders in SYS$LOGIN:[USER]MAIL.MAI;1
```

```
Press CTRL/C to cancel listing
```

```
MAIL          MEETING_MINUTES
MEMOS         PROJECT_NOTES
STAFF
```

To select a new folder as your current folder, use one of the following commands:

- **SELECT**—Selects the specified folder as the current folder.

```
MAIL> SELECT STAFF
```

```
%MAIL-I-SELECTED, 4 messages selected
```

- **DIRECTORY**—Selects the specified folder as the current folder and lists the messages in the folder.

```
MAIL> DIRECTORY STAFF
```

#	From	Date	Subject
1	BRUTUS::OSGOOD	12-NOV-1986	Staff meeting
2	BRUTUS::OSGOOD	19-NOV-1986	Staff meeting
3	CAESAR::USER	20-NOV-1986	Goodbye Sarah
4	BRUTUS::OSGOOD	3-DEC-1986	Staff meeting

- **READ**—Selects the specified folder as the current folder and displays the specified message (by default, the first message in the folder).

```
MAIL> READ STAFF
```

```
#1                12-NOV-1986  14:12:28                STAFF
```

```
From: BRUTUS::OSGOOD
```

```
To: JULIUS::USER
```

```
Subj: Staff meeting
```

```
The weekly staff meeting was cancelled due to the  
conference in Denver.
```

```
MAIL>
```

### 1.10.6 Using a Mail Subdirectory

When you receive mail messages longer than three blocks, they are written to files named MAIL\$xxxxxxxxx.MAI located in your SYS\$LOGIN directory. To avoid the display of these MAI files in your SYS\$LOGIN directory, use the SET MAIL\_DIRECTORY command, which creates a mail subdirectory and moves all your MAI files to that subdirectory. (The SHOW MAIL\_DIRECTORY command displays the name of the subdirectory that contains all your MAI files.) To move the MAI files from a subdirectory back to your SYS\$LOGIN directory, use the SET NOMAIL\_DIRECTORY command.

### 1.10.7 Using the Mail Keypad

You can use the keypad to execute commands in the Mail Utility. Most of the keypad keys execute two commands. To enter the top command shown in the following diagram, you simply press the key. To enter the bottom command shown in the following diagram, you press the PF1 key before you press the key (just as you do when using the editing keypad).

PF1 GOLD	PF2 HELP DIR/FOLDER	PF3 EXTRACT/MAIL EXTRACT	PF4 ERASE SELECT/MAIL
7 SEND SEND/EDIT	8 REPLY REP/EDIT/EXT	9 FORWARD FORWARD/EDIT	— READ/NEW SHOW/NEW
4 CURRENT CURRENT/EDIT	5 FIRST FIRST/EDIT	6 LAST LAST/EDIT	, DIR/NEW DIR MAIL
1 BACK BACK/EDIT	2 PRINT PRINT/PR/NOT	3 DIR DIR/ST=99999	ENTER  SELECT
0 NEXT NEXT/EDIT	• FILE DELETE		

7K-1744-84

To execute the MAIL command SEND, for example, press the keypad key 7. To execute the MAIL command SEND/EDIT, press the PF1 key first and then press key 7. (For more information on mail keypad commands, see Appendix MAIL).

You can redefine the keypad keys to execute MAIL commands when you are in the Mail Utility. Defining keypad keys in MAIL is similar to defining keypad keys to execute DCL commands; see the DEFINE/KEY command in Appendix MAIL for specifications.

## Chapter 2

# Storage and Output of Data

You can save data by storing it on disks and magnetic tapes, which are structured so that the data resides in discrete files. You can examine data by displaying files on the terminal screen and printing files on paper.

### 2.1 Devices

Devices are classified as either mass storage or record oriented.

Mass storage devices provide a way to save the contents of files on a magnetic medium. Files thus saved can be accessed at any time and updated, modified, or reused. Disks and magnetic tapes are mass storage devices. When disk packs or reels of magnetic tape are mounted on drives, they are called volumes.

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, mailboxes, and card readers are record-oriented devices. (Printers and card readers are also called unit record devices.)

You can use physical, logical, or generic names to refer to devices.

Note that certain commands such as `ALLOCATE` and `MOUNT` require that you specify a device name. When a command requires a device name, specify only the device portion of the file specification.



## 2-2 Storage and Output of Data

### 2.1.1 Physical Device Names

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

`ddcu`

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

The controller designation and unit number identify the location of the 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 as part of a file specification, terminate it with a colon (:).

When you refer to a file on a disk volume set, you can 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. If you do not specify a logical or physical device name, your current default device name is supplied.

### 2.1.2 Logical Device Names

Your system manager can set up logical names to indicate the devices available to you. You should use these logical names, rather than the physical device names, when referring to files. In this way, you can achieve file and device independence. If you specify a file using a logical device name, you can access the file, regardless of which physical device holds the disk or tape containing your file. Your system manager will ensure that the logical device names are always equated to the correct physical devices. When you use a logical device name in a file specification, you must terminate the name with a colon.

For example, the following file specification uses a logical device name to specify the device containing the disk volume with the file `[NOAH]ANIMALS.LIS`:

`USE1:[NOAH]ANIMALS.LIS`

As long as the system manager defines the logical name `USE1` correctly, the system can access your files, regardless of where the volume is mounted.

The VAX/VMS system also offers a special type of logical device name called a concealed device name. If a device has a concealed device name, the logical name (not the physical device name) will be displayed in system messages that refer to the device.

### 2.1.3 Generic Device Names

With the commands `ALLOCATE` and `MOUNT`, the VAX/VMS system allows you to specify generic device names in which the controller and/or the unit number is not specified. When you use a generic device name, the system locates an available controller or 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 `ALLOCATE` and `MOUNT`, the system goes to controller A if you omit the controller designation; it goes to unit number 0 if you omit the unit number.

## 2.2 File Operations on Disks

VAX/VMS disks are structured to let you create and access files through a hierarchical organization. The higher levels of the hierarchy (all but the end levels) are specially structured files called directories; the end levels are the data or program files. The access path (file specification) to a file is through the device name (identified by the logical name of the disk or the drive on which the volume is mounted), through a first-level directory, through any subdirectories, and then to the file.

For example, a disk named `CLERK` might have a first-level directory named `LICENSES`. This directory might have subdirectories named `AUTO`, `DOG`, and `MARRIAGE`, and a file named `SUM1986.DAT;18`. The `DOG` subdirectory might contain a file named `1986.DAT;32`. The access paths to the two end files are as follows:

`CLERK=> LICENSES=> SUM1986.DAT;18`

`CLERK=> LICENSES=> DOG=> 1986.DAT;32`

### 2.2.1 File Specifications

You identify a file by specifying its access path in the following format (the colon, brackets, periods, and semicolon are required delimiters):

`device:[directory.subdirectory.etc]filename.type;version`

**NOTE:** Directory specifications are valid only on disks; magnetic tapes do not have directory structures.

## 2-4 Storage and Output of Data

Field	Meaning
device	The logical name assigned to the disk or magnetic tape device containing the file (or the name, logical or physical, of the drive on which it is mounted)
directory	The name of a first-level directory; 1-39 alphanumeric characters
subdirectory	The names of up to 7 subdirectories (not including the first level directory, separated by periods; 1-39 alphanumeric characters)
filename	The name of a file; 1-39 characters; can be null
type	The file type; 1-39 characters; can be null
version	The version of the file; a positive integer in the range 1-32767; you can refer to version numbers in a relative manner by specifying 0 as the latest (highest numbered) version of the file, -1 as the next most recent version, -2 as the version before that and so on

You can use any alphanumeric character, the underscore ( \_ ), the hyphen ( - ), and the dollar sign ( \$ ) in file names and file types; however, file names and types must begin with an alphanumeric character.

The following examples demonstrate the format of a file specification.

CLERK:[LICENSES]SUM1984.DAT;18

disk	directory	file	type	version

CLERK:[LICENSES.DOG]1984.DAT;32

subdirectory

### 2.2.1.1 Setting Device and Directory Defaults

You can establish the device name, or the device plus directory name (the first-level directory and any number of subdirectories), as the first part of future file specifications by setting your default to that device and/or directory. The default remains in effect until you enter another SET DEFAULT command.

The following example displays the file CLERK:[LICENSES.DOG]1982.DAT;32.

```
$ SET DEFAULT CLERK:[LICENSES.DOG]
$ TYPE 1982.DAT;32
```

Subdirectories are specified by concatenating the subdirectory name to the name of the directory one level above it. The following example displays the same file displayed in the preceding example.

```
$ SET DEFAULT CLERK:[LICENSES]
$ TYPE [.DOG]1982.DAT;32
```

A default device and directory remains in effect until you enter another SET DEFAULT command. You can display the current default device and directory by entering the command SHOW DEFAULT. If a new default specification is incomplete, only that portion specified is replaced.

```
$ SHOW DEFAULT
CLERK: [LICENSES]
$ SET DEFAULT [MEMOS]
$ SHOW DEFAULT
CLERK: [MEMOS]
```

If you include only subdirectories as the default, they are appended to the current default.

```
$ SHOW DEFAULT
CLERK: [LICENSES]
$ SET DEFAULT [.DOG]
$ SHOW DEFAULT
CLERK: [LICENSES.DOG]
```

If you specify only the device name, the new default includes the current directory.

```
$ SHOW DEFAULT
CLERK: [LICENSES]
$ SET DEFAULT WORKDISK:
$ SHOW DEFAULT
WORKDISK: [LICENSES]
```

### 2.2.1.2 Using Default File Specifications

In most DCL commands, the file version defaults to the latest (highest numbered) version for an existing file and to either a version number of 1 or the existing version number plus 1 for a new file. Exceptions are noted in the command descriptions in Appendix DCL, as are the default file types taken by certain commands.

When a DCL parameter consists of a list of files, missing fields in a file specification default to the corresponding fields in the preceding file specification, as demonstrated:

```
$ COPY CLERK: [LICENSES.DOG] 1984.DAT, 1982 1980S.DAT
```

This example copies the latest versions of CLERK:[LICENSES.DOG]1984.DAT and CLERK:[LICENSES.DOG]1982.DAT to the file 1980S.DAT in the default directory. The second file specification (output) parameter receives as defaults the corresponding fields of the first file specification.

When you want to specify the default file type (as in the example that follows), be sure to omit the period (which would indicate a null file type).

## 2-6 Storage and Output of Data

You can revert to the default directory specification in a parameter list by preceding the file name with a null directory name. The following example copies the files CLERK:[LICENSES.DOG]1984.DAT and CLERK:[LICENSES]1982.DAT to CLERK:[LICENSES]SUM1984.DAT. Note that the second (output) parameter uses the default directory, not the directory in the first input file specification (exceptions, such as the RENAME command, are noted in the command descriptions in Appendix DCL).

```
$ SET DEFAULT CLERK:[LICENSES]  
$ COPY [.DOG]1984.DAT,[ ]1982 SUM1984
```

### 2.2.1.3 Matching Wildcards (\* and %)

With some DCL commands, you can substitute an asterisk for all or part of a field in a file specification. You can also use a percent sign for any single character in a file specification. The command then applies to all files that satisfy the portion of the file specification entered. Some examples follow.

```
$ TYPE CLERK:[LICENSES.DOG]*.*;*
```

Displays all files in the subdirectory CLERK:[LICENSES.DOG].

```
$ TYPE CLERK:[LICENSES.DOG]*.*
```

Displays the latest versions of all files in CLERK:[LICENSES.DOG].

```
$ TYPE CLERK:[LICENSES.DOG]*.DAT
```

Displays the latest versions of all DAT files in CLERK:[LICENSES.DOG].

```
$ TYPE CLERK:[LICENSES.DOG]19*.DAT
```

Displays the latest versions of all DAT files in CLERK:[LICENSES.DOG] whose file names begin with the characters 19.

```
$ TYPE CLERK:[LICENSES.*]*.DAT
```

Displays the latest versions of all DAT files in all subdirectories one level under [LICENSES].

```
$ TYPE CLERK:[*.DOG]*.DAT
```

Displays the latest versions of all DAT files in all second-level subdirectories named [.DOG].

```
$ TYPE CLERK:[LICENSES]198%.DAT
```

Displays the latest versions of all files with a file name beginning with the digits 198 and ending in any single digit and the file type DAT in the directory [LICENSES].

For files being created or modified from other files, the wildcard character indicates the substitution of the corresponding field of the first file, as demonstrated in the following examples:

```
$ COPY *.DAT *.SAV
```



Copies the latest versions of all DAT files in your default directory to new files in your default directory with the same name but a file type of SAV.

```
$ COPY 19*.DAT [SAVE]**
```

Copies the latest versions of all DAT files in your default directory beginning with the characters 19 to new files with the same names but in the directory [SAVE].

```
$ COPY [*.DOG]**;* [SAVE.]***;
```

Copies all versions of all files in all second-level subdirectories named [.DOG] (previously created with the CREATE/DIRECTORY command) to subdirectories under [SAVE] named after the first-level directories of [.DOG].

```
$ COPY [*.DOG]**;* [SAVE.***]**;
```

Copies all files in the second-level directories named [.DOG] to two levels of subdirectories under [SAVE] named [first-level-dir.DOG]. The COPY command creates the necessary subdirectories under the named directory; however, the named directory (in this case [SAVE]) must already exist.

When used with directory names in output file specifications, at least one wildcard character (to represent the file name, file type, and version number) must follow a directory specification.

You can use the asterisk wildcard character in several places in a field of an input file specification. For example, \*19\*.DAT means a file name that begins and ends with anything, but has 19 in the middle.

You cannot use the asterisk wildcard character to specify part of a version number. The wildcard character applies to the entire field.

The percent sign substitutes for a single character. For example, 198%.DAT means the latest version of all DAT files in your default directory whose names have four characters starting with 198.

Note that you can use certain qualifiers to select files from a group of files denoted by wildcard characters; see Section 2.2.3.5 for a list of such qualifiers.

#### 2.2.1.4 Search Wildcards (... and -)

The ellipsis wildcard character permits you to search down through a hierarchy of directories, as illustrated in the following examples:

```
$ TYPE [LICENSES...]1984.DAT
```

Displays the latest versions of all files named 1984.DAT in [LICENSES] and all subdirectories under [LICENSES].

```
$ TYPE [...DOG]1984.DAT
```

Displays the latest versions of the file named 1984.DAT in the [.DOG] subdirectory under your default directory.

```
$ TYPE [...]1984.DAT
```

## 2-8 Storage and Output of Data

Displays the latest versions of all files named 1984.DAT in your default directory and all subdirectories under your default directory.

```
$ TYPE [...]1984.DAT
```

Displays the latest versions of all files named 1984.DAT in all directories on the default disk.

```
$ TYPE [...DOG...]1984.DAT
```

Displays the latest versions of all files named 1984.DAT in the [.DOG] subdirectory under your default directory, and all subdirectories under the [.DOG] subdirectory.

```
$ COPY [...]*.*;* [SAVE...] *.*;*
```

Copies all files in the current directory and all files in the subdirectories under that directory to the directory named SAVE and subdirectories under the SAVE directory. The COPY command creates the necessary subdirectories under the named directory; however, the named directory (in this case [SAVE]) must already exist.

The hyphen wildcard character permits you to search up through directories. Each hyphen moves you up one directory. You can follow the hyphens with directory and/or subdirectory names to move down the directory structure on another path. For the following examples, assume a default directory of [LICENSES.MARRIAGE].

```
$ TYPE [-]1984.DAT
```

Displays the latest version of 1984.DAT in [LICENSES]

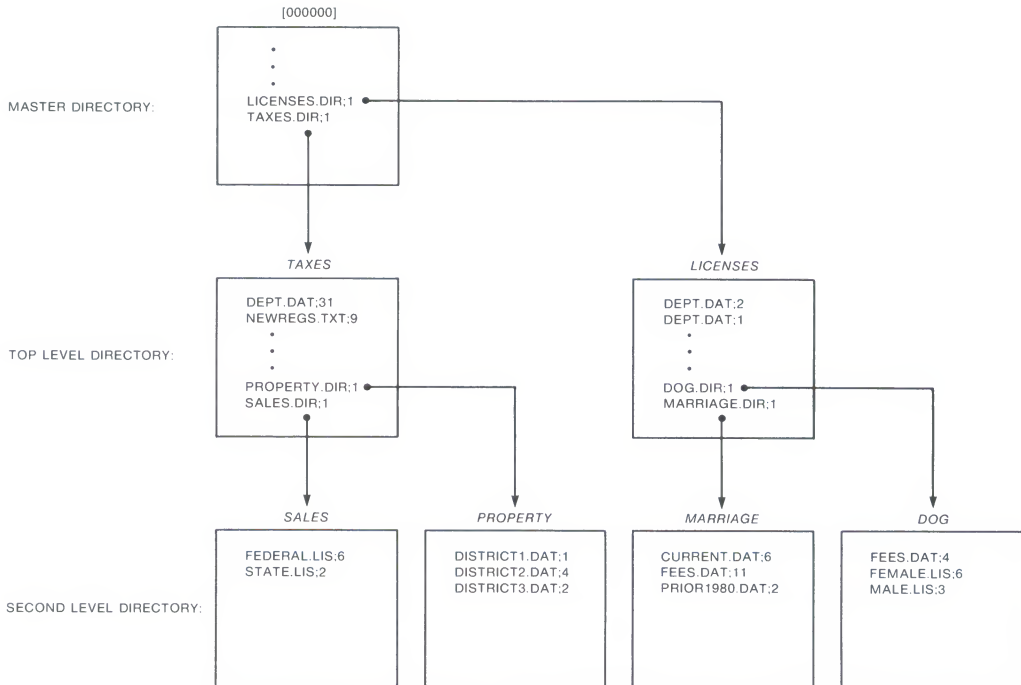
```
$ TYPE [-.DOG]1984.DAT
```

Displays the latest version of 1984.DAT in [LICENSES.DOG]

Specifying one level above a first-level directory (for example, [-], when your default directory is [LICENSES]) puts you at the level of the master directory for the disk, [000000].

### 2.2.2 Directory Operations

The master file directory (or MFD) on a disk volume exists as a file named 000000.DIR;1. Its directory name is [000000]. First-level directories exist as files named directory.DIR;1 in the master file directory. For example, the directory [LICENSES] exists as a file named LICENSES.DIR;1 in [000000]. Subdirectories exist as files named subdirectory.DIR;1 in their respective directories or subdirectories. For example, the subdirectory [LICENSES.DOG] exists as a file named DOG.DIR;1 in the directory [LICENSES]. Subdirectory levels are limited to a depth of 7; that is, the maximum number of levels, including the first-level directory (but not the master directory), is 8. The file type and version number of a directory file are always DIR and 1 respectively. In the following figure, LICENSES and TAXES are first-level directories.



ZK-1746-84

### 2.2.2.1 Displaying Directories

The DIRECTORY command displays the names of the files in a directory. The following example lists the files in [LICENSES].

```
$ SET DEFAULT CLERK: [LICENSES]
$ DIRECTORY/COLUMNS=1
```

```
Directory CLERK: [LICENSES]
```

```
AUTO.DIR;1
DOG.DIR;1
MARRIAGE.DIR;1
SUM1984.DAT;38
SUM1985.DAT;3
```

```
Total of 5 files.
```

The display shows us that [LICENSES] contains three subdirectories—[LICENSES.AUTO], [LICENSES.DOG], and [LICENSES.MARRIAGE]—and data files [LICENSES]SUM1981.DAT;38 and [LICENSES]SUM1985.DAT;3. The next example (assuming no change in the default directory) lists the contents of the subdirectory [LICENSES.DOG].

## 2-10 Storage and Output of Data

```
$ DIRECTORY/COLUMNS=1 [.DOG]
```

```
Directory CLERK:[LICENSES.DOG]
```

```
1984.DAT;38
```

```
1984.DAT;37
```

```
BITE.LIS;18
```

```
Total of 3 files.
```

To display the names of the first-level directories and files, specify [000000] as the name in the DIRECTORY command or search up one level from a first-level directory. Remember that nine of the files contained in [000000] are the structure files, and must not be deleted (see Section 2.2.4).

### 2.2.2.2 Creating Directories

The CREATE/DIRECTORY command creates a directory. The following example creates a first-level directory. (Note that you must have SYSPRV privilege to create a first-level directory.)

```
$ SET DEFAULT CLERK:
```

```
$ CREATE/DIRECTORY [LICENSES]
```

The next example creates a subdirectory.

```
$ SET DEFAULT CLERK:[LICENSES]
```

```
$ CREATE/DIRECTORY [.DOG]
```

The CREATE/DIRECTORY command creates the named directory as a file with a file type of DIR. For example, DOG.DIR is the file that corresponds to the subdirectory [.DOG]. A directory name consists of a top-level directory name plus up to seven concatenated subdirectory names enclosed in square brackets. Each name can be from 1 through 39 characters and can contain any special characters that file specifications can contain.

### 2.2.2.3 Deleting Directories

Before deleting a directory or subdirectory, make sure that it contains no files by entering the DIRECTORY command.

```
$ SET DEFAULT [LICENSES.DOG]
```

```
$ DIRECTORY
```

```
No files found.
```

You cannot delete a directory containing files. If the directory contains any files, copy or rename them to another directory (if you want to save them) and delete them from the directory of interest. If the directory contains subdirectories, check those subdirectories (copying and deleting their files) and delete the subdirectories.

To delete a directory, delete the file containing the directory (the file with the type DIR in the directory one level above the one being deleted) with the DELETE command. A directory file is created without delete access to prevent accidental deletion of the directory. Therefore, you must change the file protection to allow delete access before you can delete the directory file (see Section 7.2 for more information about file protection). The following example deletes the subdirectory [LICENSES.DOG].

```
$ SET DEFAULT [LICENSES]
$ SET PROTECTION=OWNER:D DOG.DIR
$ DELETE DOG.DIR;1
```

Remember that the directory files (files with the type DIR) for first-level directories are under the master directory and may require SYSPRV privilege to delete.

```
$ SET DEFAULT [000000]
$ SET PROTECTION=OWNER:D LICENSES.DIR
$ DELETE LICENSES.DIR;1
```

To simplify the deletion of directories, you can create a command procedure that changes the protection of the directory file and then deletes it.

```
! [MAINTENANCE]DELDIR.COM
!
! Deletes a directory.
$ ! P1 = directory name
$ SET PROTECTION=OWNER:D 'P1'.DIR
$ DELETE 'P1'.DIR;1
```

Once you have created the command procedure, you can create a symbol to invoke it.

```
$ D_DIRECTORY == "@WORK1:[MAINTENANCE]DELDIR"
```

The following commands will now delete the subdirectory [LICENSES.DOG].

```
$ SET DEFAULT [LICENSES]
$ D_DIRECTORY DOG
```

Refer to Chapter 6 for more information about command procedures.

### 2.2.3 File Operations

At DCL level, you normally deal with sequential, variable-length text files, although some commands permit access to indexed files. Your application programs may permit you to deal with files with other characteristics.



## 2-12 Storage and Output of Data

### 2.2.3.1 File Characteristics

A file consists of records, each of which consists of a number of bytes of data. The primary file characteristics are as follows:

- File organization—Sequential, indexed, or relative.

The records of a sequential file are arranged one after another in the order of creation. Records must be read from the file in order. The file must be rewritten (that is, another file or version of the file must be created) to update it.

The records of an indexed file are arranged randomly and accessed through one or more indexes. An index contains a portion of each record called a key; the keys are arranged in sequence from lowest to highest (by binary, numeric, or ASCII value depending on data type); one key is called the primary key. You can read a record directly (randomly) by specifying an index and the value of one of its keys. You can read records sequentially by specifying an index—records are read in ascending sequence according to the key values for that index, starting with the current record. You update an indexed file in place by adding, deleting, or changing records. Indexed files require more space since, in addition to the data, the indexes must be stored.

The records of a relative file are arranged in fixed-length, numbered cells. The cell numbers are used to determine the position of the record in the file. As with indexed files, you can read records sequentially or randomly. Typically, relative files are created and accessed by programs, rather than from DCL command level.

- Record format—Fixed length, variable length, or variable length with fixed control area (VFC). All records in a fixed-length file are the same size. Records in a variable-length file vary in size. Records in a VFC file have a fixed-length header followed by a variable part. Note that VFC record format is not applicable for indexed files.
- Data type—Strictly speaking, a file does not have a data type, because programs processing a file must know how each item in the file is to be interpreted. However, a file whose records contain all character data (each item is one byte interpreted according to ASCII conventions) is called a text, or character, file. A file whose data is formatted as integers, floating-point numbers, object code, or other non-ASCII data is called a binary file.
- Carriage control—New line (also known as “implied,” “carriage return,” or “CRLF”), FORTRAN carriage control, none, or print. New line places a carriage return and line feed at the end of each record when it is displayed or printed. FORTRAN carriage control uses the first character of each record to specify carriage-control information. “None” does not place carriage-control characters into a file; if you want to include control characters in the file, you must specify them as part of the data in the file. Note that the PRINT and TYPE commands

interpret carriage-return, line-feed, and form-feed characters embedded in records. Print carriage control interprets the two bytes of each VFC record as prefix and postfix carriage-control information.

Files you create using the editor or the CREATE command use new-line carriage control. Each time you press RETURN, you create a new record. When the file is printed or typed, each record appears on a new line. Files you create using the OPEN, WRITE, and CLOSE commands use print carriage control. Each WRITE command adds a new record (in VFC format) to the file.

- File size—The size of a sequential file with fixed-length records can be calculated by multiplying the number of records and the size of each record. Variable-length records require two extra bytes per record, and indexed files require space for the indexes. In addition to the files themselves, the VAX/VMS system uses disk space to store directory entries, file headers, and other file-maintenance information.

You can examine a file's characteristics with the /FULL qualifier of the DIRECTORY command.

**\$ DIRECTORY/FULL 1984.DAT**

Directory CLERK: [LICENSES.DOG]

```
1984.DAT;38                File ID: (103,75,0)
Size:      64/66           Owner:   [200,200]
Created:   02-JAN-1984 17:47 Revised: 24-MAR-1984 11:28
Expires:   <None specified> Backup:  <No backup done>
File organization: Sequential
File attributes:  Allocation=66, Extend=0
                  Global Buffer Count = 0
Record format:   Variable length
Record attributes: Carriage return
File protection: System:RWED, Owner:RWED, Group:RWED, World:RE
Access Control List None
```

Total of 1 file, 64/66 blocks.

The file size of the preceding example indicates that 64 blocks have been used out of the 66 allocated (file size is the number of actual blocks used of the blocks that have been allocated). If you are only interested in the size of the file (or several files), use the /SIZE qualifier. The following example lists the number of blocks used by the files in one directory.

## 2-14 Storage and Output of Data

**\$ DIRECTORY/SIZE**

Directory CLERK: [LICENSES.DOG]

1984.DAT;38                      64

1984.DAT;37                      62

BITE.LIS;18                      4

Total of 3 files, 130 blocks.

### 2.2.3.2 Creating and Modifying Files

The most versatile interactive tool for creating and modifying files is the interactive editor. Other tools include the following DCL commands.

- **CREATE**—Creates a text file from the lines you enter following invocation of the command:

**\$ CREATE POUND.LIS**

Tag #23, Elmer Doolittle, notified

Tag #37, Arlene Theriault, notified

No tag, light brown, 30 lbs., looks part beagle

**CTRL/Z**

Pressing CTRL/Z signals the end of the file and returns you to DCL command level. You cannot modify a file with the CREATE command. Also, you cannot modify a record in the file you are creating once you have pressed RETURN.

- **COPY**—Creates a new file from the contents of an old file. The following example copies FORMAT.TXT to WATER.TXT in the default directory.

**\$ COPY FORMAT.TXT WATER**

The COPY command can duplicate many files at a time. The following example copies all TXT files in the default directory to another directory.

**\$ COPY \*.TXT;\* [SAVETEXT]\*.\*;\***

Refer to Section 2.2.1.3 for more information about using wildcard characters in file specifications.

The COPY command can concatenate files. The following example appends WATER2.TXT to WATER1.TXT (forming a new version of WATER1.TXT) in your default directory.

**\$ COPY WATER1.TXT,WATER2.TXT WATER1.TXT**

See Appendix DCL for the full set of options and more examples.

- **RENAME**—Gives the file a new name, optionally locating it in a different directory. The following example gives the file FORMAT.TXT the new name WATER.TXT and moves it from the default directory to another directory.

**\$ RENAME FORMAT.TXT;3 [SAVETEXT]WATER.TXT**

Note that after being renamed, the file FORMAT.TXT;3 no longer exists in the default directory.

You can also create files by writing to files the information normally displayed on the terminal by TYPE, SHOW, and other commands. To do this, assign SYS\$OUTPUT to a file before entering the display command. (See Section 2.3.4 for information about system-created logical names.) Specify the /USER\_MODE qualifier with the display command so that SYS\$OUTPUT is deassigned immediately after the command executes.

```
$ DEFINE/USER_MODE SYS$OUTPUT DEVICES.LIS
$ SHOW DEVICES
```

In this example, the display produced by SHOW DEVICES goes to a new text file named DEVICES.LIS in your default directory, rather than to your terminal. You can manipulate this data as you can any text file. Most commands have an /OUTPUT qualifier for this function.

```
$ DIRECTORY/FULL/OUTPUT=FULL.LIS
```

The display produced by DIRECTORY goes to a new text file named FULL.LIS in your default directory.

You can also modify text files with the OPEN, CLOSE, READ, and WRITE commands, which are normally used in command procedures. To create a file, you open it for output, perform one or more write operations, and close it. Each write operation produces one record.

```
$ OPEN/WRITE POUND POUND.LIS
$ WRITE POUND "Tag #23, Elmer Doolittle, notified"
$ WRITE POUND "Tag #37, Arlene Theriault, notified"
$ WRITE POUND "No tag, light brown, 30 lbs."
$ CLOSE POUND
```

In the OPEN command, you specify a logical name (in this case, POUND) for the file and then use that logical name in the WRITE and CLOSE commands. The preceding sequence of commands creates a new file named POUND.LIS in the default directory (consisting of three records).

To read a text file, you open it for input, perform one read operation for each record that you want to read, and close the file. The read operation equates the value of the record to a symbol of your choice.

```
$ OPEN POUND POUND.LIS
$ READ POUND POUND1
$ CLOSE POUND
$ SHOW SYMBOL POUND1
POUND1 = "Tag #23, Elmer Doolittle, notified"
```

## 2-16 Storage and Output of Data

To update a text file, you must open the file for input and also open a file for output. Read each record from the old file and follow the instructions below for the action you wish to take.

- No change—write the symbol to the new file.
- Change—ignore the symbol and write the new contents to the new file.
- Deletion—ignore the symbol and write nothing.
- Insertion—process the last current record before locating the new record, and then write the new record to the new file before reading the next record.

The following example creates a new version of POUND.LIS, changing the third record and adding a fourth record.

```
$ OPEN POUNDOLD POUND.LIS ! Open existing file
$ OPEN/WRITE POUNDNEW POUND.DAT ! create new file
$ READ POUNDOLD POUND ! no change
$ WRITE POUNDNEW POUND
$ READ POUNDOLD POUND ! no change
$ WRITE POUNDNEW POUND
$ READ POUNDOLD POUND ! new contents
$ WRITE POUNDNEW "No tag, Charles Crowley, notified"
$ READ POUNDOLD POUND ! additional record
%RMS-E-EOF, end of file detected

$ WRITE POUNDNEW "Tag #14, Sam White, notified"
$ CLOSE POUNDOLD
$ CLOSE POUNDNEW
```

When you open a file for output (/WRITE) and the specified file already exists, a new version is created. If you specify an existing file by version number (for example, OPEN/WRITE POUNDNEW POUND.LIS;38), an error condition will result unless you specify the /APPEND qualifier with the OPEN command. An end-of-file condition occurs when you try to read beyond the last record in the file. Normally, you would perform operations such as the above in a loop in a command procedure (see Chapter 6).



### 2.2.3.3 Examining Files

The following facilities allow you to examine files on line:

- TYPE command—Displays a file on the terminal. The following example displays DOGS.DAT.

```
$ TYPE DOGS.DAT
```

If more than one file is listed in the TYPE command, the files are displayed in the order specified; if wildcard characters are used, the files are displayed in alphabetical order.

To stop the scrolling of the text on the screen temporarily, press the NO SCROLL key (F1 on VT200-series terminals); to resume scrolling, press the NO SCROLL (F1) key again. To stop the display and return to DCL command level, press CTRL/Y or CTRL/O.

- Interactive editor with /READ\_ONLY qualifier—Permits you to use interactive editing commands to move around in a file, and search for specific sequences of characters. (The /READ\_ONLY qualifier prevents you from modifying the file as you display it.) Control characters are displayed rather than being interpreted when you use /READ\_ONLY, however. For example, the form-feed character (hexadecimal 0C) appears as <FF> rather than producing a form feed.

### 2.2.3.4 Deleting Files

The DELETE command deletes files and releases the disk blocks they occupy for use by other files. The command requires you to specify a version number or the asterisk wildcard character in each file specification, as shown in the following examples:

```
$ DELETE POUND.LIS;17
```

Deletes version 17 of POUND.LIS.

```
$ DELETE POUND.LIS;16, POUND.LIS;17
```

Deletes versions 16 and 17 of POUND.LIS.

```
$ DELETE POUND.LIS;*
```

Deletes all versions of POUND.LIS.

When you delete many files with wildcard characters, you should confirm each deletion by specifying the /CONFIRM qualifier.

```
$ DELETE/CONFIRM *.*;*
CLERK: [LICENSES.DOG]1981.DAT;38, delete? [N]:
CLERK: [LICENSES.DOG]BITE.LIS;18, delete? [N]:
CLERK: [LICENSES.DOG]BITE.LIS;17, delete? [N]:
CLERK: [LICENSES.DOG]POUND.LIS;19, delete? [N]:
```

## 2-18 Storage and Output of Data

Similarly, you may want to display the names of files as they are deleted; specify the /LOG qualifier with the DELETE command.

```
$ DELETE/LOG *.LIS;*
_%DELETE-I-FILDEL, CLERK: [LICENSES.DOG]BITE.LIS;18 deleted (38 blocks)
_%DELETE-I-FILDEL, CLERK: [LICENSES.DOG]BITE.LIS;17 deleted (35 blocks)
_%DELETE-I-FILDEL, CLERK: [LICENSES.DOG]POUND.LIS;19 deleted (5 blocks)
```

The PURGE command deletes old versions of a file or files, as demonstrated in the following examples:

```
$ PURGE
```

Deletes all but the latest version of each file in the default directory.

```
$ PURGE/KEEP=2
```

Deletes all but the latest two versions of each file in your default directory.

```
$ PURGE WATER.TXT
```

Deletes all but the latest version of WATER.TXT in your default directory.

Purging sequential files after updating them enables you to retain more free space on your disk volumes.

### 2.2.3.5 Common Qualifiers

Typically used with a command that includes a wildcard character in its file specification, the following qualifiers select files from the group of files specified by the wildcard characters. You can use these qualifiers with most commands that manipulate files, including the APPEND, BACKUP, COPY, DELETE, PURGE, RENAME, and TYPE commands:

- /BEFORE[=time]—Selects files dated prior to a particular time. Specify one of the following qualifiers with /BEFORE to indicate the date to be used as the basis for selection:

Qualifier	Date
/BACKUP	The last backup
/CREATED	Creation
/EXPIRED	Expiration
/MODIFIED	The last modification

You can specify time as absolute or as a combination of absolute and delta times, or as one of the following keywords: TODAY, TOMORROW, and YESTERDAY. Refer to Section 5.4 for information about specifying the date and time.

- `/BY_OWNER[=uic]`—Selects files owned by the user with the specified user identification code (UIC). You can specify either the numeric or alphanumeric forms of the UIC. (Note that the BACKUP command accepts the `/OWNER_UIC` qualifier instead of the `/BY_OWNER` qualifier.)
- `/CONFIRM`—Requests confirmation for each file operation, processing only those files which you confirm by typing Y. This qualifier is especially useful for saving files marked for deletion by the DELETE and PURGE commands.
- `/EXCLUDE(=file-spec,...)`—Prevents the specified files from being affected by the command. Wildcard characters are allowed in the file specification.
- `/SINCE[=time]`—Selects files dated after a particular time. Specify one of the following qualifiers with `/BEFORE` to indicate the date to be used as the basis for selection:

Qualifier	Date
<code>/BACKUP</code>	The last backup
<code>/CREATED</code>	Creation
<code>/EXPIRED</code>	Expiration
<code>/MODIFIED</code>	The last modification

You can specify time as absolute or as a combination of absolute and delta times, or as one of the following keywords: TODAY, TOMORROW, and YESTERDAY. Refer to Section 5.4 for information about specifying the date and time.

In the following example the `/SINCE` qualifier selects for copying only the files in the directory [83ACCOUNTS] that have been modified since MAY 15, 1983.

```
$ COPY/SINCE=15-MAY-1983/MODIFIED [83ACCOUNTS]*.*
```

## 2.2.4 System Directories and Files

The system maintains several special directories for system files. A number of first-level files on each volume are reserved for structure information: INDEXF.SYS;1 (index file), BITMAP.SYS;1 (storage bit map), BADBLK.SYS;1 (bad block file), 000000.DIR;1 (master directory), CORIMG.SYS;1 (core image file), VOLSET.SYS;1 (volume set list file), CONTIN.SYS;1 (continuation file), BACKUP.SYS;1 (backup log file), BADLOG.SYS;1 (pending bad block log file), and QUOTA.SYS;1 (disk quota file, which exists only if disk quotas are enabled). These files are created at initialization time and updated as the volume is used.

In addition, the system disk contains a large number of files for the images, command procedures, and data required to run the operating system. These files are in the directories [SYSEXE], [SYSHLP], [SYSLIB], [SYSMGR], and [SYSMSG], located under the system directory [SYS0] on the system disk. You can refer to these directories by placing their names after the logical name SYS\$SYSROOT, for example, SYS\$SYSROOT:[SYSEXE]. In addition, these directories have special logical names as described in Section 2.3.

### 2.3 Logical Names

A logical name is a name that you can use in place of a file specification, part of a file specification, or another logical name. Logical names serve two main functions:

- **Shorthand and readability**—You can define commonly used files, directories, and devices with short, meaningful logical names. Such names are easier to remember and type than the full file specifications. Names that you use frequently can be defined in your login command procedure. Names that most users on your system use frequently can be defined in the site-specific system startup command procedure.
- **File independence**—You can define logical names to associate input/output operations with the appropriate files for use in images and command procedures. For example, if a command procedure references the logical name ACCOUNTS, you can equate ACCOUNTS to any file on any disk before executing the command procedure. (System images—commands and utilities—generally use predefined logical names (see Section 2.3.4) so that you need not explicitly define them. Logical names defined in MOUNT commands provide device independence.

The system maintains logical names in logical name tables.

#### 2.3.1 Logical Name Definition

Once you have equated a logical name to one or more equivalence strings, you can use the logical name to reference those equivalence strings.

### 2.3.1.1 Defining Logical Names

The DEFINE command creates a logical name and associates it with one or more equivalence names. The following example associates the logical name ACCOUNTS with the equivalence name WORK1984:[ACCOUNTS].

```
$ DEFINE ACCOUNTS WORK1984:[ACCOUNTS]
```

You can now use the logical name ACCOUNTS in commands to mean the equivalence name WORK1984:[ACCOUNTS]. If the logical name is part of a file specification, it must be the first (left) part and must be separated from the rest of the file specification by a colon. The following example displays the file PAYROLL.DAT in the directory [ACCOUNTS] on the disk WORK1984.

```
$ TYPE ACCOUNTS:PAYROLL.DAT
```

If the equivalence name represents the name of a device (whether it is a physical device name or is itself a logical name), terminate the equivalence name with a colon in the DEFINE command. The following command equates the logical name ACCOUNTS\_DISK with the device name WORKDISK.

```
$ DEFINE ACCOUNTS_DISK WORKDISK:
```

The following commands may also be used to define logical names.

- ALLOCATE—Allocates a device for exclusive use and optionally assigns a logical name for that device.
- ASSIGN—Associates one or more equivalence names with a logical name.
- MOUNT—Mounts a volume and assigns a logical name for the volume.
- OPEN—Associates the specification of the file being opened with a logical name. (The function of the logical name, in this case, is to identify the file in subsequent READ, WRITE, and CLOSE commands.)

You can equate more than one logical name with an equivalence string. For example, you can equate the logical names \$TERMINAL and CONSOLE to the physical name of a terminal so that both logical names translate to the same device. (If you equate a logical name to more than one equivalence string in a single command, you create a search list for the system to use to translate the names; see Section 2.3.2.4 for information about search list translation.)

If you equate a logical name to one equivalence string and then equate the same logical name to another equivalence string, the second definition supersedes the first. You can, however, equate the same logical name to different equivalence strings if the logical name definitions are in different tables (see Section 2.3.3.1). You can equate the same logical name to different strings in the same table if they are defined in different access modes (see Section 2.3.3.4).



## 2-22 Storage and Output of Data

Use care when defining a logical name with an equivalence string that is identical to a component of a file specification. For example, suppose that a user-created image exists as a file named CLERK:[USEREXE]DOG.EXE with the requirement that a logical name DOG be defined before running the image. You enter the following commands with the intent of defining DOG and running the image.

```
$ DEFINE DOG CLERK1:[LICENSES.DOG]1984.DAT
$ SET DEFAULT CLERK:[USEREXE]
$ RUN DOG
```

You do not execute CLERK:[USEREXE]DOG.EXE as intended because the system translates DOG to be CLERK1:[LICENSES.DOG]1984.DAT.

If you cannot access a file, and the command you are specifying and the file specification seem in order, check the left-hand component of the file specification (with SHOW LOGICAL) to be sure that it is not defined as a logical name.

### 2.3.1.2 Deassigning Logical Names

Logical names in process-private tables and your job table are automatically deleted when your process terminates. (Section 2.3.3 discusses logical name tables.)

User-mode logical names in your process table are automatically deleted following the execution of a single image. The following commands explicitly deassign logical names.

- DEASSIGN—Deassigns the logical name defined interactively.
- DISMOUNT—Deassigns the logical name defined when the volume was mounted.
- CLOSE—Deassigns the logical name defined when the file was opened.

Note that any logical name defined through the ALLOCATE command is not automatically deassigned when the device is deallocated. You must deassign it explicitly with the DEASSIGN command.

### 2.3.1.3 Displaying Logical Names

The SHOW LOGICAL command displays the iterative translations of a logical name.

```
$ SHOW LOGICAL ACCOUNTS_DISK
"ACCOUNTS_DISK" = "WORKDISK" (LNM$PROCESS_TABLE)
```

In this display, LNM\$PROCESS\_TABLE refers to the logical name table in which the definition resides (see Section 2.3.3.1).

### 2.3.2 Logical Name Translation

You can substitute a logical name for a file specification or for the leftmost components of a file specification. In the latter case, terminate the logical name with a colon when you use it as a reference. The equivalence name for the specified logical name takes the place of the omitted components. The following examples all display the file WORKDISK:[ACCOUNTS]1984.DAT.

```
$ DEFINE ACCOUNTS WORKDISK: [ACCOUNTS] 1984.DAT
$ TYPE ACCOUNTS
$ DEFINE ACCOUNTS_FILE WORKDISK: [ACCOUNTS] 1984
$ TYPE ACCOUNTS_FILE: .DAT
$ DEFINE ACCOUNTS_DIR WORKDISK: [ACCOUNTS]
$ TYPE ACCOUNTS_DIR: 1984.DAT
$ DEFINE ACCOUNTS_DISK WORKDISK:
$ TYPE ACCOUNTS_DISK: [ACCOUNTS] 1984.DAT
```

Note that if you combine a logical name with an explicitly stated file type or version number only, you must include the period or semicolon, respectively. For example, if ACCOUNTS is equivalent to DUA1:[ACCOUNTS]1984.DAT, ACCOUNTS:2 is an invalid file specification. (ACCOUNTS;;2 is valid.) Defaults for the current directory, the file type (depending on the function being performed), and the version number are applied as usual after translation.

#### 2.3.2.1 Iterative Translation

You can associate a logical name with one or more equivalence names that are themselves logical names or that contain logical names. Most system commands and utilities translate file specifications iteratively (up to 10 times) until all logical names are resolved. In the following example, ACCOUNTS is translated to WORKDISK: with the SHOW LOGICAL command.

```
$ DEFINE ACCOUNTS_DISK WORKDISK:
$ DEFINE ACCOUNTS ACCOUNTS_DISK
$ SHOW LOGICAL ACCOUNTS

"ACCOUNTS" = "ACCOUNTS_DISK" (LNM$PROCESS_TABLE)
1 "ACCOUNTS_DISK" = "WORKDISK:" (LNM$PROCESS_TABLE)
```

Some functions (as noted in the appropriate descriptions) do not translate logical names iteratively. The SHOW TRANSLATION command, for example, provides only the immediate equivalence name.

```
$ DEFINE ACCOUNTS_DISK WORKDISK:
$ DEFINE ACCOUNTS ACCOUNTS_DISK
$ SHOW TRANSLATION ACCOUNTS
ACCOUNTS = "ACCOUNTS_DISK" (LNM$PROCESS_TABLE)
```

### 2.3.2.2 Noniterative Translation

To prevent an equivalence name from being translated iteratively, define it with the `/TRANSLATION_ATTRIBUTES=TERMINAL` qualifier. The VAX/VMS system does not attempt to translate the equivalence string of a logical name defined with the `TERMINAL` attribute. The following example equates the logical name `WORKDISK` to the physical name `DUA0`. If `DUA0` is later defined as a logical name, the translation of `WORKDISK` will nevertheless stop at `DUA0`.

```
$ DEFINE/TRANSLATION_ATTRIBUTES=TERMINAL WORKDISK DUA0
```

### 2.3.2.3 Concealed Translation of Device Names

A concealed device name causes the logical name for the device to be shown in system displays rather than the physical name to which the system actually translates the logical name. To define a device name so that it is concealed in system displays (except for the `SHOW LOGICAL` display), specify the `/TRANSLATION_ATTRIBUTES=CONCEALED` qualifier with the `DEFINE` command.

```
$ DEFINE/TRANSLATION_ATTRIBUTES=CONCEALED WORKDISK DUA1
```

Device names are concealed by default with the `MOUNT` command. You can display the concealed device name by specifying the associated logical name in a `SHOW LOGICAL` command, or by referring to the device name rather than the logical name (for example, in a `DIRECTORY` command); otherwise, the system uses the logical name in displays.

### 2.3.2.4 Search List Translation

You can specify more than one equivalence string for a logical name in a single `DEFINE` command. The order in which you specify the equivalence strings determines the order in which the system translates the names. The following example defines the logical name `ACCOUNTS` to be both `WORK1983:[ACCOUNTS]` and `WORK1984:[ACCOUNTS]`. The subsequent `DIRECTORY` command searches the `ACCOUNTS` directory on the disk named `WORK1983` first, and then the `ACCOUNTS` directory on the disk named `WORK1984`.

```
$ DEFINE ACCOUNTS WORK1983:[ACCOUNTS],WORK1984:[ACCOUNTS]  
$ DIRECTORY ACCOUNTS:ACME.DAT
```

```
Directory WORK1983:[ACCOUNTS]
```

```
ACME.DAT;3
```

```
Total of 1 file.
```

```
Directory WORK1984:[ACCOUNTS]
```

```
ACME.DAT;1
```

```
Total of 1 file.
```

```
Grand total of 2 directories, 2 files.
```

If you use a search list with a DCL command that does not accept wildcard characters in a file specification, the VAX/VMS system uses the search list to translate only until it finds a valid file specification; that is, the command affects only the first file found. For example, the RUN command (which does not accept wildcard characters) would execute only one program in the following example (and, if no file TAX.EXE exists, would generate a single file-not-found error message).

```
$ DEFINE ACCOUNTS WORK1983: [ACCOUNTS] ,WORK1984: [ACCOUNTS]
$ RUN ACCOUNTS: TAX. EXE
```

### 2.3.3 Scope and Precedence of Logical Names

The VAX/VMS system maintains logical names and their equivalence strings in tables that are either shareable (across multiple processes) or private to the process that created them. Of the four default logical name tables (LNM\$PROCESS, LNM\$JOB, LNM\$GROUP, and LNM\$SYSTEM), process is process-private, group and system are shareable across multiple processes, and job is shareable among the process and subprocesses in a single job tree.

Logical name tables and the logical name definitions they contain can be:

- **Process-private**—Available only to your process and deassigned when you log out. You should define logical names at this level unless you have a definite need for other processes to use them. By default, the DEFINE command creates process-private logical names. Process-private logical names are defined in the process logical name table LNM\$PROCESS (also called LNM\$PROCESS\_TABLE). You can define process-private logical names in your LOGIN.COM command procedure, so that they will be defined for you each time you log in.
- **Shareable**—Available to all processes (or to all processes with the same group number) in the system. System logical names are usually set up in SYSTARTUP.COM and are deleted when the system shuts down. Use the /SYSTEM (or /GROUP) qualifier to define a logical name in the system logical name table LNM\$SYSTEM (or group logical name table, LNM\$GROUP); SYSNAM (or GRPNAM) privilege is required. (Logical names that are shared among your login process and its subprocesses—but not among other processes—are defined in the job logical name table LNM\$JOB.)



### 2.3.3.1 Logical Name Table Directories

The system maintains the names of logical name tables in logical name table directories. The names of logical name tables are logical names that either define tables or translate iteratively to table names; they can be up to 31 alphanumeric (including the dollar sign and underscore) characters in length. The names of logical name tables are contained in one of the following directory tables:

- **Process directory table**—By default, the process logical name table directory (LNM\$PROCESS\_DIRECTORY) contains the following logical names when you log in. Some of the logical names identify logical name tables; others translate iteratively to logical name tables. All tables catalogued in the process logical name table directory are private to the process that created them.

Logical Name	Translation
LNM\$PROCESS_DIRECTORY	The process directory table
LNM\$PROCESS_TABLE	The process logical name table; takes precedence; the default
LNM\$PROCESS	A logical name for the process logical name table (equivalent to LNM\$PROCESS_TABLE)
LNM\$GROUP	A logical name for your group logical name table (equivalent to LNM\$GROUPxxx)
LNM\$JOB	A logical name for your process tree logical name table (equivalent to LNM\$JOBxxx)

- **System directory table**—By default, the system logical name table directory (LNM\$SYSTEM\_DIRECTORY) contains the following logical names when you log in. Some of the logical names identify logical name tables; others translate to logical name tables. All tables catalogued in the system logical name table are shareable.

Logical Name	Translation
LNM\$SYSTEM_DIRECTORY	The system directory table
LNM\$SYSTEM_TABLE	The system logical name table; requires SYSNAM or SYSPRV privilege to create or delete logical names contained in it
LNM\$SYSTEM	A logical name for the system logical name table (equivalent to LNM\$SYSTEM_TABLE)
LNM\$GROUP_xxx	The group logical name table, where xxx is the group number; requires GRPNAM or SYSNAM privilege to create or delete a logical name contained in it



Logical Name	Translation
LNМ\$JOBxxx	The job logical name table, where xxx is a unique number for each job tree
LNМ\$DCL_LOGICAL	A logical name for the search list specified by LNМ\$FILE_DEV; the list is LNМ\$PROCESS, LNМ\$JOB, LNМ\$GROUP, LNМ\$SYSTEM; the DCL commands SHOW LOGICAL and SHOW TRANSLATION use LNМ\$DCL_LOGICAL to determine the tables to be searched
LNМ\$DIRECTORIES	A logical name for the following: LNМ\$PROCESS_DIRECTORY and LNМ\$SYSTEM_DIRECTORY
LNМ\$FILE_DEV	A logical name for the following search list: LNМ\$PROCESS, LNМ\$GROUP, LNМ\$JOB, LNМ\$SYSTEM; used by all DCL commands to translate logical names used to locate files
LNМ\$PERMANENT_MAILBOX	A logical name that translates iteratively to the logical name table in which logical names associated with permanent mailboxes are entered
LNМ\$TEMPORARY_MAILBOX	A logical name that translates iteratively to the logical name table in which logical names associated with temporary mailboxes are entered

Generally, you do not need to change the default logical name table definitions set up in the directory tables, LNМ\$PROCESS\_DIRECTORY and LNМ\$SYSTEM\_DIRECTORY. Two reasons for changing the entries in the directory tables are: (1) to create another logical name table and (2) to change the search order for file specification logical names by redefining LNМ\$FILE\_DEV.

To display the names defined in a particular logical name table, you can use the /TABLE qualifier. For example, to display the contents of the system logical name table, specify:

```
$ SHOW LOGICAL/TABLE=LNМ$SYSTEM
```

Note that to display the process, job, group, or system tables, you can specify the corresponding qualifier: /PROCESS, /JOB, /GROUP, and /SYSTEM. The following command has the same effect as the preceding example.

```
$ SHOW LOGICAL/SYSTEM
```

To display an entry in one of the directory tables, you must specify the /TABLE qualifier. The following example displays the equivalence strings for the logical name LNМ\$DCL\_LOGICAL.

```
$ SHOW LOGICAL/TABLE=LNМ$SYSTEM_DIRECTORY LNМ$DCL_LOGICAL
```

### 2.3.3.2 Precedence

Identical logical names can exist in more than one table. The logical name that is used depends on the order in which the logical name tables are searched. For example, when the system attempts to translate a logical name in order to identify the location of a file, it uses the logical name LNM\$FILE\_DEV to provide the list of tables in which to look for the name. The order in which the tables are listed is also the order in which they are searched. The precedence order defined by LNM\$FILE\_DEV is: (1) process table, (2) job table, (3) group table, (4) system table. Therefore, if a logical name exists in both the process and the group logical name tables, the logical name within the process table is used. The following examples demonstrate the use of the same logical name in process and system tables.

```
$ DEFINE/SYSTEM ACCOUNTS WORKDISK:[ACCOUNTS]1984.DAT
```

Defines ACCOUNTS as a system logical name.

```
$ DEFINE ACCOUNTS WORKDISK:[ACCOUNTS]1982.DAT
```

Defines ACCOUNTS as a process logical name.

```
$ SHOW LOGICAL ACCOUNTS
```

```
"ACCOUNTS" = WORKDISK:[ACCOUNTS]1982.DAT (LNM$PROCESS_TABLE)
```

Displays the logical name that takes precedence.

```
$ SHOW LOGICAL/SYSTEM ACCOUNTS
```

```
"ACCOUNTS" = "WORKDISK:[ACCOUNTS]1984.DAT" (LNM$SYSTEM_TABLE)
```

Displays the system logical name.

```
$ DEASSIGN ACCOUNTS
```

Deletes the process logical name.

```
$ SHOW LOGICAL ACCOUNTS
```

```
"ACCOUNTS" = "WORKDISK:[ACCOUNTS]1984.DAT" (LNM$SYSTEM_TABLE)
```

Displays the logical name that takes precedence.

Multiple tables with the same name may also exist. For example, there may exist both a process-private and a shareable table called MY\_TABLE. The process-private version always takes precedence over the shareable table in all logical name table processing. When a logical name, such as LNM\$FILE\_DEV, is used as a table name, the logical name is iteratively translated until a list of table names is formed. During this iterative translation, each name is first translated in the process directory and, if this translation fails, it is then translated in the system directory. This order of precedence cannot be changed. As a consequence of this ordering, a logical name placed in the process directory table for use as a table name will always take precedence over any identical name residing in the system directory.

### 2.3.3.3 Logical Name Table Creation

The CREATE/NAME\_TABLE command creates a logical name table and catalogs it in one of the directory logical name tables. (Logical names that identify logical name tables or that translate iteratively to logical name tables must always be entered into one of the directory logical name tables.) To create a logical name table that is private to your process, create the table in LNM\$PROCESS\_DIRECTORY (the default). If you want the table to be shareable, specify /PARENT\_TABLE=LNMSYSTEM\_DIRECTORY with the CREATE/NAME\_TABLE command. Creating shareable name tables requires SYSPRV privilege or ENABLE access to the parent table.

The following example creates a process-private logical name table named TAX, places the definition for the logical name CREDIT in the table, and verifies the table's creation. (You must specify the /TABLE qualifier with the SHOW LOGICAL command to display a logical name in any table other than LNM\$SYSTEM or LNM\$PROCESS.)

```
$ CREATE/NAME_TABLE TAX
$ DEFINE/TABLE=TAX CREDIT [ACCOUNTS.1984]CREDIT.DAT
$ SHOW LOGICAL/TABLE=TAX CREDIT

"CREDIT" = "[ACCOUNTS.1984]CREDIT.DAT" (TAX)
```

To make the system search a user-created logical name table automatically when processing file specifications, you must create a process-private version of the default search list (LNM\$FILE\_DEV) in LNM\$PROCESS\_DIRECTORY. To add the created table's name to the default search list, you can define LNM\$FILE\_DEV as follows:

```
$ DEFINE/TABLE=LNMS$PROCESS_DIRECTORY LNM$FILE_DEV -
_$ TAX, LNM$PROCESS, LNM$JOB, LNM$GROUP, LNM$SYSTEM
```

Placing the table's name first specifies that the system search that table first, and so on in the order of specification.

To delete a logical name table, specify the table that contains it (the system or process directory logical name table) and the name of the table. Deleting a shareable logical name table requires DELETE access to the table or SYSPRV privilege. For example, to delete the logical name table TAX of the preceding example, specify the following command line:

```
$ DEASSIGN/TABLE=LNMS$PROCESS_DIRECTORY TAX
```

Note that all logical names in descendant tables (and the descendant tables themselves) are deleted when a parent logical name table is deleted.

#### 2.3.3.4 Access Modes

A logical name has an associated access mode of user, supervisor, executive, or kernel. The default access mode for the DEFINE command is supervisor; you must have SYSNAM privilege to create an executive mode logical name.

You can equate the same logical name to different equivalence strings in the same logical name table by specifying different access modes for each definition. The following example equates the directory named ACCOUNTS to two different strings in the process logical name table—one in supervisormode (the default) and one in usermode.

```
$ DEFINE ACCOUNTS WORKDISK: [ACCOUNTS] 1984.DAT
$ DEFINE/USER_MODE ACCOUNTS WORKDISK: [ACCOUNTS] 1984.DAT
```

Define a logical name in usermode when you want to define it only for the execution of the next DCL or user image. In the following example, the logical name ACCOUNTS is automatically deleted after the execution of PAYABLE.

```
$ DEFINE/USER_MODE ACCOUNTS WORKDISK: [ACCOUNTS] 1984.DAT
$ RUN PAYABLE
```

In looking up logical names, all privileged images and utilities, such as LOGINOUT and MAIL, bypass the user- and supervisor-mode portions of the system logical name table (LNM\$SYSTEM\_TABLE). Therefore, DIGITAL recommends that logical names for important system components (public disks and directories, for example) be defined in *executive mode*, using the DCL command DEFINE/SYSTEM/EXECUTIVE. This operation requires either the SYSPRV or SYSNAM privilege.

### 2.3.4 System-Created Logical Names

The system creates a number of logical names for you when you start the system and when you log in.

#### 2.3.4.1 Process-Permanent Logical Names

The following table lists the logical names automatically created for each process when you log in. You cannot deassign these logical names. You can redefine them (by specifying the same name in a DEFINE command), but if the redefined name is later deassigned, the process-permanent name is reestablished. These logical names are available to each user of the system at the process level.

Logical Name	Equivalence Name Interactive mode	Batch mode
SYS\$INPUT (default input stream)	Terminal <sup>1</sup>	Disk <sup>2</sup>
SYS\$COMMAND (original input stream)	Terminal <sup>1</sup>	Disk <sup>2</sup>
SYS\$OUTPUT (default output stream)	Terminal <sup>1</sup>	Disk <sup>3</sup>
SYS\$ERROR (default error message output)	Terminal <sup>1</sup>	Disk <sup>3</sup>

<sup>1</sup>Terminal: your terminal

<sup>2</sup>Disk: initial input device

<sup>3</sup>Disk: initial output device

### 2.3.4.2 System-Permanent Logical Names

The following table lists the logical names automatically defined when the system starts up. These names are available to all users of the system at the system level.

Logical Name	Equivalence Name
DBG\$INPUT	SYS\$INPUT at the process level
DBG\$OUTPUT	SYS\$OUTPUT at the process level
SYS\$ERRORLOG	SYS\$SYSROOT:[SYSERR]
SYS\$HELP	SYS\$SYSROOT:[SYSHLP]
SYS\$LIBRARY	SYS\$SYSROOT:[SYSLIB]
SYS\$MAINTENANCE	SYS\$SYSROOT:[SYSMAINT]
SYS\$MANAGER	SYS\$SYSROOT:[SYSMGR]
SYS\$MESSAGE	SYS\$SYSROOT:[SYSMSG]
SYS\$NODE	Name of your node if you are on a network
SYS\$SHARE	SYS\$SYSROOT:[SYSLIB]
SYS\$SYSDEVICE	System disk (usually SYS\$DISK)
SYS\$SYSROOT	SYS\$SYSDEVICE:[SYS0.]
SYS\$SYSTEM	SYS\$SYSROOT:[SYSEXE]
SYS\$TEST	SYS\$SYSROOT:[SYSTEST]
SYS\$UPDATE	SYS\$SYSROOT:[SYSUPD]

In general, you do not need to redefine process-permanent and system-permanent logical names. Occasionally, you may wish to redefine SYS\$INPUT or SYS\$OUTPUT to redirect terminal input/output. The best practice is to define them with the /USER\_MODE qualifier so that they will automatically revert to their initial definitions after the next image runs (see Section 2.3.3.4).



## 2.4 Printing Files

To print a file or files, use the PRINT command.

The PRINT command places your print job (all the files to be printed) in a list of jobs to be printed called the print queue. The following example places a print job containing three files in the default print queue, SYS\$PRINT.

```
$ PRINT POUND, DOGS.DAT, CAT
```

Job POUND (queue SYS\$PRINT, entry 202) started on SYS\$PRINT

The system displays the job name (POUND), the queue name (SYS\$PRINT), the job number (202), and indicates whether the job has started or is pending. Once a job is submitted, you reference it using the job number. The file types of the files named in the PRINT command default to LIS or the last explicitly named file type; thus, the preceding example queues POUND.LIS, DOGS.DAT, and CAT.DAT to SYS\$PRINT. Once the job is queued, it will be printed when no other jobs precede it in the queue and when the printer is physically ready to print.

A print queue can execute only one job at a time. Print jobs are scheduled for printing according to their priority, and the job with the highest priority is printed first. If more than one job exists with the same priority, the smallest job is printed first (unless the queue was initialized with the /SCHEDULE=NOSIZE qualifier). Jobs of equal size having the same priority are selected for printing according to their submission time, where the job submitted earliest is printed first.

The default print queue, SYS\$PRINT, is usually initialized and started as part of the site-specific system startup procedures. The SHOW QUEUE command displays the queues that have been initialized and the status of your jobs; specify the /ALL qualifier to see jobs queued for other users. In the following example job POUND is the only job in the queue SYS\$PRINT.

```
$ SHOW QUEUE/ALL SYS$PRINT
```

Printer queue SYS\$PRINT

Jobname	Username	Entry	Blocks	Status
-----	-----	-----	-----	-----
POUND	USER	202	38	Printing

By default, the job name is the name of the first (or only) file specified in the PRINT command. Use the job entry number to delete the job from the queue.

```
$ DELETE/ENTRY=202 SYS$PRINT
```

## 2.4.1 Controlling a Print Job

Various commands and qualifiers allow you to control print jobs in the following ways (see Appendix DCL for the complete specifications of each command and additional qualifiers).

Print Operations	Print Job Commands and Qualifiers
Number of copies	
By job	PRINT/JOB_COUNT=n <sup>1</sup>
By file	PRINT/COPIES=n <sup>1</sup>
Specified file only	file-spec/COPIES=n <sup>1</sup>
Number of pages	PRINT/PAGES=1
Print features	
Flag pages	PRINT/FLAG=1
Type of forms (paper)	PRINT/FORM= 1
Special features	PRINT/CHARACTERISTICS=1
Double-spacing	PRINT/SPACE <sup>1</sup>
Page heading	PRINT/HEADER <sup>1</sup>
Notification of job execution	PRINT/NOTIFY
Delaying execution of a job	
For a specified time	PRINT/AFTER
Indefinitely	PRINT/HOLD
Releasing a delayed job	SET QUEUE/RELEASE/ENTRY
Stopping a print job	
Delete job	DELETE/ENTRY=
Stop currently printing job and begin printing the next job in the queue	STOP/ABORT
Stop currently printing job and requeue it for printing	STOP/REQUEUE

<sup>1</sup>Parallel qualifiers for the SET QUEUE/ENTRY command allow you to specify these operations for print jobs that are already queued but not yet printing.

Operations that affect print jobs are checked against the R (read) and D (delete) protection specified for the queue and the owner UIC of the job.

## 2.4.2 Controlling a Print Queue

You must have OPER privilege or have EXECUTE access to a queue to act as an operator of that queue. See Appendix DCL for access and privilege restrictions on individual commands described in this section.

### 2.4.2.1 Stopping and Restarting a Queue

Print queues are usually initialized and started as part of the system startup procedure. However, in some situations you may need to stop and start a queue from a running system. After the system has been booted, the START/QUEUE /MANAGER command must be issued before any other queue commands can be entered. For example, you may need to stop a queue to change forms or to fix the printer. The steps involved in stopping and restarting a print queue are as follows:

1. Enter the STOP/QUEUE/NEXT command—To stop a queue temporarily (without deleting it). The /NEXT qualifier allows the current job to finish printing before the queue is stopped. If the /NEXT qualifier is omitted, the current job is interrupted and the queue “pauses” instead of stopping.  
\$ STOP/QUEUE/NEXT SYS\$PRINT
2. Wait for current jobs to complete—See the /NEXT qualifier in the previous step.
3. Do whatever is necessary to the printer—Change forms (paper) or change the ribbon, for example.
4. Enter the START/QUEUE command—To restart a print queue that has been previously initialized.  
\$ START/QUEUE SYS\$PRINT

### 2.4.2.2 Creating a Print Queue

If you are creating a new print queue, you should spool the printer and initialize the queue before starting it. (Spooling uses secondary storage to buffer files passing between slow I/O devices, such as line printers, and the programs that generate them. This allows the system to produce output as quickly as the disk device allows instead of at printer speed. When the printer file is closed by the program, the temporary file spooled to disk is queued for printing.) Spool the printer as shown:

```
$ SET DEVICE/SPOOLED=SYS$PRINT $TERMINAL1
```

Use the DCL command INITIALIZE/QUEUE/START to initialize and start the queue. (The following example assumes that the systemwide logical name \$PRINTER has been assigned to the print device.)

```
$ INITIALIZE/QUEUE/START/ON=$PRINTER SYS$PRINT
```

If you initialize and start a printer with one command, you can queue and print jobs immediately (unless an incompatible print characteristic or form was specified in the PRINT command). After a queue has been started, you can stop and restart it (with

STOP/QUEUE and START/QUEUE respectively) without affecting the jobs in the queue.

#### 2.4.2.3 Specifying Print Features and Restrictions

The qualifiers you use when you initialize or start a queue allow you to specify default print features for the queue and to control the jobs that can execute on the queue.

- **Default print features**—To assign default print features to a queue, specify the /DEFAULT qualifier with the INITIALIZE/QUEUE, START/QUEUE, or SET QUEUE commands and one or more of the following keywords:

BURST	Burst page printed preceding output
FEED	Form feed inserted at end of page
FLAG	Flag page printed preceding output
FORM=type	Default form for a printer, terminal, or server queue
TRAILER	Trailer page printed following output

- **Print queue restrictions**—To restrict the types of jobs that can execute on the queue, specify one of the following qualifiers with the INITIALIZE/QUEUE, START/QUEUE, or SET QUEUE commands:

/BLOCK_LIMIT	Restricts by size
/CHARACTERISTIC	Restricts by characteristics
/FORM	Restricts by form (paper)

#### 2.4.2.4 Using the FORM=type Keyword

You can change the default form type for a specific output queue by using the /DEFAULT=FORM=type option to the DCL command INITIALIZE/QUEUE, START/QUEUE, or SET QUEUE. The new default form must first be defined using the DEFINE/FORM=type command. (See the *VAX/VMS DCL Dictionary* for a description of the DEFINE/FORM command.)

The stock (type of paper stock) of the default form must be identical to the stock of the mounted form, or any job submitted without an explicit form will enter a pending state until they are made identical.

To find out the default form for a particular queue, issue the DCL command SHOW QUEUE/FULL *queue-name*. The following example shows the default form for a queue named LN03\$PRINT.

```
$ SHOW QUEUE/FULL LN03$PRINT
```

```
Printer queue LN03$PRINT, on ALPHA::ALPHA$LCA0, mounted form STANDARD (stock=8x10)
/BASE_PRIORITY=4 /DEFAULT=(FEED,FLAG,FORM=REPORT(stock=8x10),TRAILER=ONE)
/NOENABLE_GENERIC Lowercase /OWNER=[1,4] /PROTECTION=(S:E,O:D,G:R,W:W)
```

In this example, the default form is named REPORT and the stock is named 8x10. The stock of the default form matches the stock of the mounted form. All jobs not associated with an explicit form definition will be associated with the form named REPORT by default. Since the stock of the form STANDARD matches the stock of the default form REPORT, all jobs submitted to this queue without an explicit form definition will be processed.

### 2.4.3 Using Multiple Print Queues

The number and type of queue that you need depends on the number of printers your system has and your particular print needs (for instance, whether and how often you need special print forms). Printer queues can be one of the following types of queues:

- Printer queue—A queue assigned to a specific print device.
- Terminal queue—A printer queue assigned to a hardcopy terminal that is being used solely as a printer (not interactively).
- Generic queue—A queue that distributes the processing of jobs to printers with similar characteristics. Jobs submitted to a generic queue are held in that queue until one of the assigned printer queues becomes available.

#### 2.4.3.1 One Printer with Multiple Queues

If you want to be able to queue certain jobs (low-priority jobs or jobs requiring special print forms, for example) for printing during off-peak hours, you can initialize a queue without starting it. Jobs sent to a queue that is initialized but not started will not print until you stop the other queue and start this queue. For example, the following commands initialize and start the first queue, SYS\$PRINT, and initialize but do not start the second queue, LOW\_PRIORITY. The first PRINT command prints the file REG\_FILE.DAT on SYS\$PRINT; the second PRINT command queues but does not print the file BIG\_FILE.DAT.

```
$ INITIALIZE/QUEUE/ON=$PRINTER SYS$PRINT/START
$ INITIALIZE/QUEUE/ON=$PRINTER LOW_PRIORITY
$ PRINT REG_FILE.DAT
Job REG_FILE (queue SYS$PRINT, entry 202) started on SYS$PRINT.
$ PRINT BIG_FILE.DAT/QUEUE=LOW_PRIORITY
Job BIG_FILE (queue LOW_PRIORITY, entry 3) pending.
```

To start the second queue so that the jobs queued to it can print, do the following:

```
$ STOP/QUEUE/NEXT SYS$PRINT
$ SHOW QUEUE SYS$PRINT
Printer queue SYS$PRINT, stopped
$ START/QUEUE LOW_PRIORITY
```



The STOP/QUEUE/NEXT command stops the queue SYS\$PRINT after the current job finishes printing; use the SHOW QUEUE command to be sure that the job has finished printing. The START/QUEUE command starts the queue LOW\_PRIORITY so that jobs previously queued to it can print.

### 2.4.3.2 One Queue with Multiple Printers

If your system has more than one printer of a similar type, you should initialize at least one generic queue. Because the PRINT command queues jobs to SYS\$PRINT by default, you should initialize SYS\$PRINT as the generic queue with the INITIALIZE/QUEUE/GENERIC command in the system startup procedure, specifying the generic queues with the /GENERIC qualifier. Queues assigned to printers that are remote, that use special forms, or that possess unique printer characteristics should be explicitly disabled as generic queues with the /NOENABLE\_GENERIC qualifier, to prevent them from being assigned to the generic queue by default. A terminal queue assigned to a generic queue should be initialized and started with the /TERMINAL qualifier.

In the following example, the print queues PRINTER\_1 and PRINTER\_2 are initialized and then assigned to the generic queue SYS\$PRINT.

```
$ INITIALIZE/QUEUE/ON=LPAO: PRINTER_1/START
$ INITIALIZE/QUEUE/ON=LPBO: PRINTER_2/START
$ INITIALIZE/QUEUE SYS$PRINT/START/GENERIC=(PRINTER_1,PRINTER_2)
```

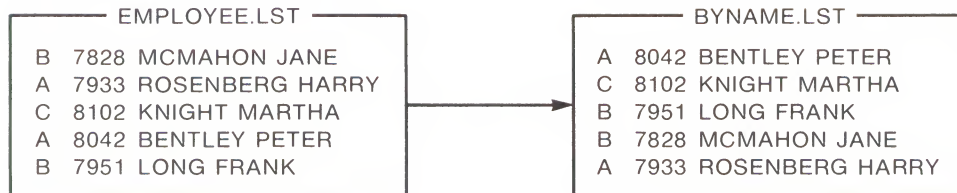
## 2.5 Sorting and Merging Files

The SORT and MERGE commands let you combine and reorder files at the DCL command level. The SORT command reorders records in a file (or files) so that they are in alphabetic or numeric order, either low to high (ascending) or high to low (descending), according to a portion of each record called the key. The MERGE command combines up to 10 sorted files into one ordered output file.

### 2.5.1 Record Sorting

Record sorting, the default type of sort operation, keeps records intact and produces an output file consisting of complete records. The following example demonstrates an ascending (the default) record sort based on that portion of each record starting at character position 8 and extending to the end of the record (the name).

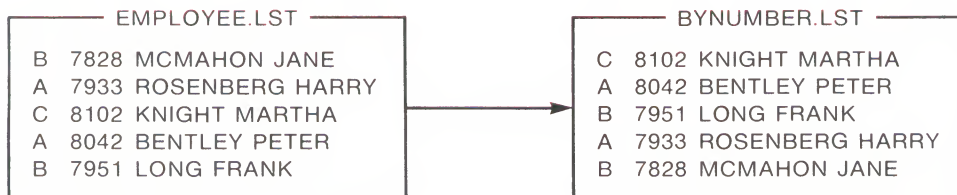
```
$ SORT/KEY=(POSITION=8,SIZE=15) EMPLOYEE.LST BYNAME.LST
```



ZK-1748-84

The next example sorts the same file in descending order using positions 3 through 6 (the number) as the key.

**\$ SORT/KEY=(POSITION=3,SIZE=4,DESCENDING) EMPLOYEE.LST BYNUMBER.LST**



ZK-1749-84

The first parameter of the SORT command names the file or files to be sorted. Multiple files are treated as one large file for sorting purposes. The second parameter provides a name for the ordered output file that the sort will create. The following example sorts the records in two files, EMPLOYEE.LST and EMPLOYER.LST, and creates the ordered output file BYNAME.LST.

**\$ SORT EMPLOYEE.LST,EMPLOYER.LST BYNAME.LST**

### 2.5.1.1 Single Key

By default, the SORT command assumes a key field that begins in the first position of a record, that includes the entire record, and that will be sorted in ascending order. Use the /KEY qualifier to specify characteristics of the key field other than those assumed by default. In the following example, the /KEY qualifier specifies that the key starts in position 8 and is 15 characters in length.

**\$ SORT/KEY=(POSITION=8,SIZE=15) EMPLOYEE.LST BYNAME.LST**

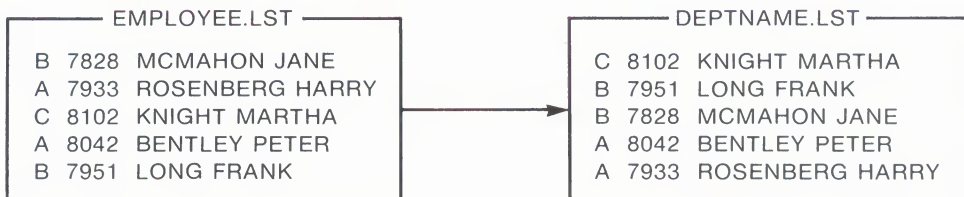
(If an actual key would have to extend beyond the end of the record to meet the size specification—for example, if the key is the last item in a variable-length format—the missing characters are treated as nulls.)

### 2.5.1.2 Multiple Keys

You can specify more than one key (up to a limit of 255) and each key can be ascending or descending. Specify multiple keys in their order of priority. For example, the following command sorts records first on the value of position 1 in descending order, then on the value of positions 8 through 27 (or the end of the record) in ascending order.

```
$ SORT/KEY=(POSITION=1,SIZE=1,DESCENDING) -
_$ /KEY=(POSITION=8,SIZE=15) -
_$ EMPLOYEE.LST DEPTNAME.LST
```

The results of the sort specified in the preceding example are as follows.



ZK-1764-84

By default, records with identical keys are kept but not sorted predictably. To retain identical keys and arrange them according to the input file order, specify the /STABLE qualifier. To eliminate duplicate keys, specify the /NODUPPLICATES qualifier.

### 2.5.2 Other Types of Sorting

In addition to record sorting, you can perform the following types of sorts.

- **Tag sort**—Sorts the keys only and then rereads the input file to produce an output file of complete records. The net result is the same as for a complete record sort. A tag sort is useful if disk space is at a premium, because it typically uses less scratch file space. Time may be saved if the records are large but the keys are relatively small. Specify the /PROCESS=TAG qualifier with the SORT command to generate a tag sort.
- **Address sort**—Sorts the keys only and produces an output file of record addresses (RFAs) in binary format. An address sort is faster than a record sort, but you must write a program to associate the record addresses with the records of the input file. Specify the /PROCESS=ADDRESS qualifier to generate an address sort.

- Indexed sort—Sorts the keys only and produces an output file of keys and record addresses (RFAs). The addresses are in binary format. An index sort is faster than a record sort, but you must write a program to associate the record addresses with the records of the input file. Specify the `/PROCESS=INDEX` qualifier to generate an index sort.

### 2.5.3 Character Data Files

The SORT command assumes by default that the files to be sorted contain character data. ASCII is the default collating sequence for character data. You can specify EBCDIC to generate an output file that is ordered in EBCDIC sequence (although it remains in ASCII representation). In general, ASCII orders numbers (0 through 9) first, then uppercase letters (A through Z), and then lowercase letters (a through z).

The multinational collating sequence collates according to the international character set defined by DIGITAL (see Appendix CHAR). The multinational collating sequence compares for different characters first, then for different diacritical forms of the same character (formed by using diacritical marks as part of “compose sequences” on VT200 series terminals), and then for different cases (uppercase or lowercase) of the same character. To use the multinational collating sequence, specify the `/COLLATING_SEQUENCE=MULTINATIONAL` qualifier.

**NOTE:** Use caution when using the multinational collating sequence to sort or merge files for further processing. Sequence checking procedures in most programming languages compare numeric characters. Because the multinational sequence is based on actual graphic characters (and not the codes representing those characters), normal sequence checking will not work.

### 2.5.4 Noncharacter Data Files

If you sort files containing items other than character data, you must specify the data type of each key. Also, you must take care in calculating starting positions and sizes, as the items being compared may occupy more than one byte. For example, if you are sorting a file that contains 20 characters followed by 3 floating-point numbers in F floating format, and the key is the last floating-point number, you must make the following specification:

```
$ SORT/KEY=(POSITION=29,F_FLOATING) STATS.RAW STATS.SOR
```

The character data occupies positions 1 through 20 (20 characters), the first F floating-point number occupies position 21 through 24, the second F floating-point number occupies positions 25 through 28, and the third F floating-point number occupies positions 29 through 32. The size of the floating-point number is not specified (since it is fixed at 4 bytes).

### 2.5.5 Terminal Input

To enter the input records for a SORT or MERGE operation from your terminal, specify SYS\$INPUT as the input file parameter, qualifying it with the size of the longest record (in bytes) and the approximate size of the input file (in blocks). After you enter the command, enter the input records. Terminate each record by pressing RETURN and terminate the file by pressing CTRL/Z. The following example demonstrates a sort operation in which the input comes from the terminal.

```
$ SORT/KEY=(POSITION=8,SIZE=15) -
_$ SYS$INPUT/FORMAT=(RECORD_SIZE=22,FILE_SIZE=10) BYNAME.LST
B 7828 MCMAHON JANE
A 7933 ROSENBERG HARRY
C 8102 KNIGHT MARTHA
A 8042 BENTLEY PETER
B 7951 LONG FRANK
[CTRL/Z]
```

### 2.5.6 Output File Organization

You must specify the file organization of the output file of a SORT or MERGE operation if that organization differs from that of the input file. Assume, for example, that EMPLOYEE.LST is an indexed file and you wish the output file produced by the sort to be a sequential file.

```
$ SORT/KEY=(POSITION=8,SIZE=15) -
_$ EMPLOYEE.LST BYNAME.LST/SEQUENTIAL
```

If the organization of the output file is indexed, the file must already exist and be empty. You must also qualify the output file parameter with /OVERLAY.

To change the format and order of the records in the output file, use a specification file; see Section 2.5.8 for information on specification files.

### 2.5.7 Batch Job Submission

If you are sorting large files, submission of the sort as a batch job is advisable since the sort will require some time. If the input is a file, the command procedure you submit must contain the SORT command and either set your default directory or include the directory in the file specifications.

```
$ SUBMIT SORTJOB
! SORTJOB.COM
!
$ SET DEFAULT [USER.PER]
$ SORT/KEY=(POSITION=8,SIZE=15)-
EMPLOYEE.LST BYNAME.LST
```



## 2-42 Storage and Output of Data

You can include the input records in the batch job by placing them after the sort command, one record per line.

```
$ SUBMIT SORTJOB
! SORTJOB.COM
!
$ SET DEFAULT [USER.PER]
$ SORT/KEY=(POSITION=8,SIZE=15)-
SYS$INPUT-
/FORMAT=(RECORD_SIZE=22,FILE_SIZE=10)-
BYNAME.LST
B 7828 MCMAHON JANE
A 7933 ROSENBERG HARRY
C 8102 KNIGHT MARTHA
A 8042 BENTLEY PETER
B 7951 LONG FRANK
```

As with terminal input, specify the input file parameter as SYS\$INPUT and qualify it with the record size (in bytes) and the approximate file size (in blocks).

### 2.5.8 Specification Files

Specification files allow you to:

- Create or modify a collating sequence
- Reformat records in the output file
- Conditionally select key and data fields

Create a specification file with the CREATE command or the editor; the default file type of a specification file is SRT. List in the file the qualifiers and keywords that define the sort or merge operation you desire. (See the /SPECIFICATION qualifier of the SORT or MERGE commands in Appendix DCL for a complete list of the qualifiers allowed in a specification file.) The order of qualifiers matters only when you are sorting and specifying more than one key field, when you are describing output format, and when you are defining multiple record types. (In these cases, the order in which you specify qualifiers is the order in which they are processed.) Begin comments with an exclamation point.

To use a specification file, include the /SPECIFICATION qualifier in the SORT or MERGE command line as follows:

```
$ SORT/SPECIFICATION=NAME_SPEC BY_NUMB.DAT BY_NAME.DAT
```

Any DCL command qualifiers override corresponding qualifiers in the specification file.

### 2.5.8.1 Creating or Modifying a Collating Sequence

You can create or modify a collating sequence by including the `/COLLATING_SEQUENCE` qualifier in the specification file.

- **Modifying a collating sequence**—To modify a collating sequence, specify the `MODIFICATION` option with the `/COLLATING_SEQUENCE` qualifier. For example, to modify the ASCII sequence so that the string "MC" is treated as the equivalent of the letter "M," include the following qualifier in a specification file.

```
/COLLATING_SEQUENCE=(SEQUENCE=ASCII,
                      MODIFICATION=("MC"="M"),
                      IGNORE=("'"),FOLD)
```

The `IGNORE` option in the preceding example specifies that the apostrophe be ignored during the sort or merge (that is, the next character is collated). The `FOLD` option specifies that uppercase and lowercase letters be treated as equivalents. (The multinational collating sequence folds lowercase letters into uppercase letters by default.) Because the `FOLD`, `MODIFICATION`, and `IGNORE` options are processed in the order in which they are specified, you should generally specify the `FOLD` option last to ensure that the `MODIFICATION` and `IGNORE` options affect uppercase and lowercase characters.

- **Creating a collating sequence**—To define a collating sequence, specify the sequence as a string of characters, single or double, or a range of single characters in the order of priority. You must specify in the sequence all characters that appear in the key field; characters not specified by the sequence are ignored (unless you specify them with the `FOLD` or `MODIFICATION` options). Enclose each character or string in quotation marks and separate them with commas. The following example defines a collating sequence that sorts only records beginning with the letters A, B, and C.

```
$ SORT/COLLATING_SEQUENCE=(SEQUENCE=("A","B","C"))
```

**2.5.8.2 Reformatting Records in the Output File**

By default, the record format of an output file is the same as that of the input file. To eliminate or reorder fields in your output file, first name the appropriate fields and then specify each field by name in a /DATA qualifier in the specification file. The order in which you specify the /DATA qualifiers is the order in which the fields appear in the output file. You must specify each field that you want to appear. For example, you could alter the following record format with the qualifiers below:

```
B 7828 MCMAHON JANE
A 7933 ROSENBERG HARRY
C 8102 KNIGHT MARTHA
A 8042 BENTLEY PETER
B 7951 LONG FRANK
```

```
/FIELD=(NAME=LETTER,POSITION=1,SIZE=1)
/FIELD=(NAME=FULLNAME,POSITION=8,SIZE=15)
/KEY=FULLNAME
/DATA=LETTER
/DATA=" "
/DATA=FULLNAME
```

```
.
.
.
```

Note that /DATA=" " in the preceding example inserts a space between the letter and surname fields of each record.

**2.5.8.3 Conditionally Selecting Key and Data Fields**

You can select fields conditionally by first defining the condition in a /CONDITION qualifier and then specifying the condition in an /INCLUDE, /OMIT, /KEY, or /DATA qualifier. Use one of the following operators to define the condition test: EQ, NE, GT, GE, LT, or LE (and optionally AND or OR). For example, to omit all records containing the letter C in the first field (LETTER) of the preceding example, you could define the condition to be the existence of the letter C in that field and then specify the /OMIT qualifier for all occurrences of that condition, as follows:

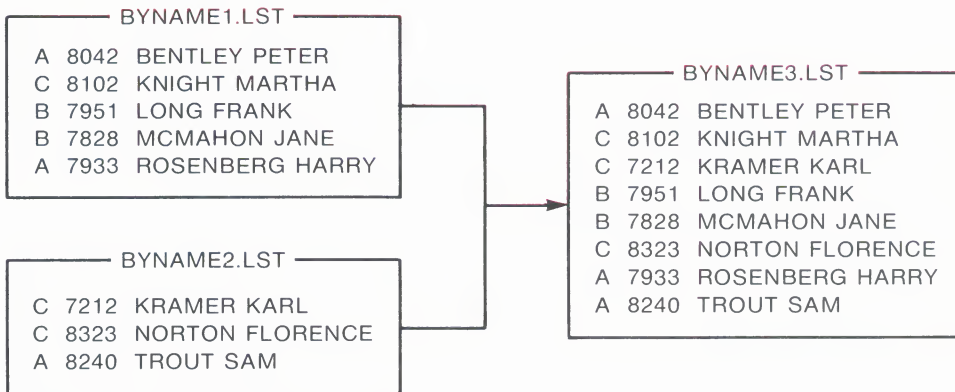
```
/FIELD=(NAME=LETTER,POSITION=1,SIZE=1)
/FIELD=(NAME=SURNAME,POSITION=8,SIZE=15)
/KEY=SURNAME
/DATA=LETTER
/DATA=SURNAME
/CONDITION=(NAME=LETTER_C,TEST=(LETTER EQ "C"))
/OMIT=(CONDITION=LETTER_C)
```

```
.
.
.
```

### 2.5.9 Merging Files

The MERGE command combines up to 10 sorted files (that is, the input files must be in order) into one ordered output file. The input files must have the same format and must have been sorted on the same key fields. The following example demonstrates the merging of two files based on that portion of each record starting at position 8 and extending to the end of the record (the name).

```
$ MERGE/KEY=(POSITION=8,SIZE=15) BYNAME1.LST,BYNAME2.LST BYNAME3.LST
```



ZK-1771-84

By default, MERGE does sequence checking to ensure the input files are in order. The sequence check stops the merge if a record is found to be out of order. To prevent sequence checking during the merge, specify the `/NOCHECK_SEQUENCE` qualifier.

## 2.6 Using Libraries

A library is a specially formatted file in which you store data in units called modules. System-defined libraries include object, shareable image, macro, help, and text libraries. Various system components process libraries. The HELP command, for example, displays information according to the contents of help libraries. Unless you are programming, you typically use only help and text libraries. (For the complete specifications of the LIBRARY command, see Appendix DCL.)

### 2.6.1 Creating Text Libraries

A text library stores modules that contain lines of text. Each text file inserted into the library corresponds to one library module. A text library has a default file type of TLB; an input file to a text library has a default file type of TXT. Specify the /TEXT qualifier with the LIBRARY command to identify a text library.

Use the /TEXT and /CREATE qualifiers to create a text library. The following command creates a text library named MEMO.TLB.

```
$ LIBRARY/TEXT/CREATE MEMO
```

When you create a library, you can optionally specify a file or a list of files that contain modules to be placed in the library. You can also exceed the default limit of 31 characters per text module name by specifying the size of the longest module name with the KEYSIZE option of the /CREATE qualifier. (See Section 2.6.2 for an example of the KEYSIZE option.)

### 2.6.2 Creating Help Libraries

A help library stores modules of specially formatted lines of text. A help source file can contain any number of modules; every level-1 line starts a new library module. A help library has a default file type of HLB; a help source file has a default file type of HLP. Specify the /HELP qualifier with the LIBRARY command to identify a help library.

Use the /HELP and /CREATE qualifiers to create a help library. When you create a library, you can optionally specify a file or a list of files that contain modules to be placed in the library. For example, the following command creates a help library named HELPLIB.HLB (first parameter) from a source help file named HELPLIB.HLP (second parameter).

```
$ LIBRARY/HELP/CREATE HELPLIB HELPLIB
```

The /CREATE qualifier allows you to exceed the default limit of 15 characters per module name by specifying the size of the longest module name with the KEYSIZE option. If the number of modules exceeds 256, specify the number (or a larger number to give yourself some leeway) with the MODULES option. The following example makes use of both options.

```
$ LIBRARY/HELP/CREATE=(KEYSIZE:19,MODULES:400) -  
_ $ HELPLIB HELPLIB
```



### 2.6.2.1 Creating Help Modules

Source files for a help library are formatted so that each module is a topic for the HELP command. Within each module, the text is arranged in levels: typing HELP level-1-name obtains all the information under the level-1 topic plus the names of the level-2 topics; typing HELP level-1-name level-2-name obtains all the information under the level-2 topic plus the names of the level-3 topics; and so on. In addition, information about qualifiers (topics whose names begin with a slash) can be obtained discretely even though the qualifiers are not numbered as topics. The name of the level-1 topic is the name of the module.

A source help file can contain any number of modules. The general format of each module is as follows:

```
1 module-name
first-level-text ...
2 topic-name
second-level-text ...
.
.      (lower-level modules as desired)
.
2 topic-name
second-level-text ...
.
.      (more second-level and lower modules as desired)
.
```

Observe the following rules in formatting modules in source help files:

- **Module levels**—Lower-level topics are logically under the higher-level topics that they physically follow. Level numbers must be one digit in column 1; no other number (for example, a number that is part of the text) must be in column 1.
- **Qualifiers**—A line beginning with a slash is treated as a separate topic at the same level as the level number it physically follows.
- **Module names**—A module name (that is, a first-level topic name) is restricted to the number of characters specified for the key size when the library is created or compressed (the default is 15 characters). In addition, the module name can only contain alphanumeric characters, the underscore, the dollar sign, and the hyphen. Extra characters, and all characters following an invalid character, are truncated from the module name. (For example, if you specify CREATE/DIRECTORY as a first-level topic, the module name will be taken as CREATE—all characters starting with the slash are truncated. If you specify CREATE\_DIRECTORY as the module name and the key size for the library is 15, the module name will be taken as CREATE\_DIRECTOR.)
- **Comments**—Lines with an exclamation point in column 1 are interpreted as comments.

You should give your source help file a file type of HLP. If the source file represents the entire contents of the library, give it the same name as the library. If the source file represents part of the contents of the library, give it a meaningful name for those contents. The following example represents a source help file named CREATE.HLP, which contains two modules.

#### CREATE.HLP

##### 1 CREATE

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

##### Format

CREATE file-spec[...]

##### 2 Parameters

file-spec

Specifications of the files being created. No wildcards are allowed.

##### 2 Qualifiers

/LOG

/NOLOG (default)

Displays the file specification of each new file created.

##### 1 CREATE\_DIRECTORY

Creates one or more new directories or subdirectories.

##### Format

CREATE/DIRECTORY directory-spec[...]

##### 2 Parameters

directory-spec[...]

Disk device (optional) and name of new directory or subdirectory. No wildcards are allowed.

##### 2 Qualifiers

/DIRECTORY

Indicates creation of a directory rather than a file. This qualifier is required.

/LOG

/NOLOG (default)

Displays the directory specification of each directory created.

### 2.6.2.2 Naming Help Libraries

If you intend the help library to be the main help library for your system, give it a file name of HELPLIB. Otherwise, give it a meaningful name that corresponds to its contents. The following example creates a help library named PAYABLE.HLB from a source help file named PAYABLE.HLP.

```
$ LIBRARY/HELP/CREATE PAYABLE PAYABLE
```

The /LIBRARY qualifier of the DCL command HELP allows you to search another library in place of the default libraries.

**CAUTION:** Place site-specific help in separate libraries that are defined as logical names (HLP\$LIBRARY\_1, HLP\$LIBRARY\_2, etc.) in SYSTARTUP.COM. Do NOT place site-specific help in the system library provided by VAX/VMS (SYS\$LIBRARY:HELPLIB.HLB) as it could interfere with VAX/VMS updates and layered-product installations.

### 2.6.3 Displaying Libraries

The /LIST qualifier displays information concerning the library and lists the modules in the library. The /LIST/FULL qualifiers list additional information on each module. The following example displays information on and the contents of the library MEMO.TLB.

```
$ LIBRARY/TEXT/LIST MEMO
```

```
Directory of TEXT library AC:[MEMOS]MEMO.TLB;1 on 15-APR-1986 16:21:41
Creation date: 10-APR-1986 21:34:40 Creator: VAX-11 Librarian V-04-00
Revision date: 10-APR-1986 21:43:46 Library format: 4.0
Number of modules: 67 Max. key length: 31
Other entries: 0 Preallocated index blocks: 7
Recoverable deleted blocks: 0 Total index blocks used: 4
Max. number history records: 20 Library history records: 4
AIR1
```

```
.
.
.
```

```
WATER3
```

You can display a history of the updates performed on the library by specifying /HISTORY with /LIST. Each history record identifies the user who performed the update, the number of modules affected, and the date of the update. A specification of /LIST/HISTORY/FULL names the modules that were inserted or deleted.

You can limit the modules displayed with the /ONLY qualifier. The following example lists only those modules whose names begin with WATER in the library MEMO.TLB.

```
$ LIBRARY/LIST/ONLY=WATER* MEMO
```

You can redirect the display to a file (the default for the output is SYS\$OUTPUT) by specifying the name of the file with the /LIST qualifier. The following example stores the library display in a file named MEMO.LIS.

```
$ LIBRARY/TEXT/LIST=MEMO/FULL MEMO
```

To examine the contents of a module or modules, use the /EXTRACT qualifier. The following example displays on SYS\$OUTPUT the contents of the module WATER2 in the library MEMO.TLB.

```
$ LIBRARY/TEXT/EXTRACT=WATER2/OUTPUT=SYS$OUTPUT MEMO
```

### **2.6.4 Deleting Libraries**

To delete a library, use the DELETE command, as shown:

```
$ DELETE MEMO.TLB;1
```

Be sure, however, that you no longer need the contents of the library or that you have converted the library back to source files with the /EXTRACT qualifier. Otherwise, you may accidentally delete a large amount of data.

### **2.6.5 Adding Library Modules**

You can add modules to a library at creation time by specifying the source files as the second parameter. (The first parameter is the name of the library.) The following example creates the library MEMO.TLB and inserts the modules named WATER1, WATER2, and AIR1 (contained in files named WATER1.TXT, WATER2.TXT, and AIR1.TXT) in the library.

```
$ LIBRARY/TEXT/CREATE MEMO WATER1,WATER2,AIR1
```

To add new modules to an existing library, use the /INSERT or /REPLACE qualifier. The /INSERT qualifier does not add a module if a module with the same name already exists in the library (instead an error message results). The /REPLACE qualifier, the default, inserts the module if the module does not already exist in the library or replaces the module of the same name if it does exist. The following example adds the module WATER3 (contained in the file WATER3.TXT) to the library MEMO.TLB, replacing any modules of the same name in the library.

```
$ LIBRARY/TEXT/REPLACE MEMO WATER3
```

## 2.6.6 Deleting Library Modules

To delete a module from a library, specify the module name (not the file name) with the /DELETE qualifier. The following example deletes the module WATER3 from the library MEMO.TLB.

```
$ LIBRARY/TEXT/DELETE=WATER3 MEMO
```

You can use a wildcard character to delete a group of modules, as shown in the following example, which deletes all modules beginning with the letters PAY from the library HELPLIB.HLB.

```
$ LIBRARY/HELP/DELETE=PAY* HELPLIB
```

Deleting modules from a library does not reduce the size of the library file. The space used by the deleted modules remains unused until new modules requiring it are added. You can recover the free space generated by the deletions by creating another library file with the /COMPRESS qualifier.

```
$ LIBRARY/HELP/DELETE=(CREATE,CREATE_DIRECTORY)-  
_$ /COMPRESS HELPLIB  
$ PURGE HELPLIB.HLB
```

The REDUCE option of the /DATA qualifier further compresses the text of the library. Libraries in reduced format require less disk space but take slightly longer to access.

## 2.6.7 Modifying Library Modules

To modify a module or modules in a text library, take the following steps:

1. Extract the module from the library with the /EXTRACT qualifier of the LIBRARY command.
2. Make the necessary edits to the module.
3. Return the modules to the library with the /REPLACE qualifier of the LIBRARY command.

The following example modifies the module WATER3 in the library MEMO.TLB.

```
$ LIBRARY/TEXT/EXTRACT=WATER3/OUTPUT=WATER3 MEMO  
$ EDIT WATER3.TXT
```

```
.  
      editing session  
.
```

```
$ LIBRARY/TEXT/REPLACE MEMO WATER3  
$ DELETE WATER3.TXT;*
```



Note that the module is named as part of the /EXTRACT qualifier and that it is the name of the module, not the file containing the module. The /OUTPUT qualifier names the file produced by the /EXTRACT operation; if you omit /OUTPUT, the file is named after the library, for example, MEMO.TLB. Once you have replaced the modules, you can delete the source file.

To make wholesale changes to a library (for example, to make a change that affects all commands in a help library), re-create a library by taking the following steps:

1. Extract all the modules. Create a source file that represents the entire contents of the library by specifying /EXTRACT=\*. Do not use the /OUTPUT qualifier. The name of the source file will be the same as the library (but the file type will be HLP or TXT). The following example extracts all the modules from HELPLIB.HLB and creates a source help file named HELPLIB.HLP from them.

```
$ LIBRARY/HELP/EXTRACT=* HELPLIB
```

2. Edit the source help file. Add, delete, and change modules in the source help file as you see fit.

3. Create a new library. Create a new library with the original name, using the edited source file as input, and purge the old library file.

```
$ LIBRARY/HELP/CREATE HELPLIB HELPLIB
```

```
$ PURGE HELPLIB.HLB
```

## 2.7 Transferring Files Between Systems

You can move files between your system and another system by prefixing file specifications with the node name of the other system and a double colon. For example, the following command displays on your terminal screen the file CLERK:[ACCOUNTS]FY83.SUM from the system whose node name is EBONY.

```
$ TYPE EBONY::CLERK:[ACCOUNTS]FY83.SUM
```

You can define a node name plus a device name, or a node name plus a device and directory name, as a logical name. The following example equates the logical name ACCOUNTS to a partial file specification and then uses the logical name in place of that portion of the file specification.

```
$ DEFINE ACCOUNTS EBONY::CLERK:[ACCOUNTS]
```

```
$ TYPE ACCOUNTS:FY83.SUM
```

If the name of the file on the other system is not a valid VAX/VMS file specification, enclose it (starting after the double colon) in quotation marks; the quotation marks prevent the local VAX/VMS system from performing syntax checking or logical name translation.

```
$ TYPE EBONY::"$FY1984.SUMMARY"
```

The SHOW NETWORK command lists the systems available to your node if it is a routing node. (If your local node is a nonrouting or end node, you will receive a message to that effect.)

### 2.7.1 Reading Files from Another System

Use the COPY command to move files from another system to your system, as demonstrated in the following examples:

```
$ COPY EBONY::CLERK:[ACCOUNTS]FY83.DAT *
```

Moves the latest version of CLERK:[ACCOUNTS]FY83.DAT on the system EBONY to a file named FY83.DAT in your default directory.

```
$ COPY EBONY::CLERK:[ACCOUNTS]*.* *
```

Moves the latest versions of all files in CLERK:[ACCOUNTS] on the system EBONY to files with the same names in your default directory.

You can also use the DIRECTORY, PRINT, READ (with OPEN and CLOSE), and TYPE commands to examine files on another system.

### 2.7.2 Writing Files to Another System

The COPY command also moves files from your system to another system, as demonstrated in the following examples:

```
$ COPY FY83.DAT EBONY::CLERK:[ACCOUNTS]
```

Moves the latest version of FY83.DAT in your default directory to a file of the same name in the directory CLERK:[ACCOUNTS] on the system EBONY.

```
$ COPY *.* EBONY::CLERK:[ACCOUNTS]
```

Moves the latest versions of all files in your default directory to files with the same names in the directory CLERK:[ACCOUNTS] on the system EBONY.

You can also use the Mail Utility to move files to another system: the file is copied to the mail file of the account rather than to a specified file. The following example copies the file FY83.DAT to USER's mail file on the system EBONY.

```
$ MAIL FY83.DAT EBONY::USER
```

You can print files on another system by copying files to spooled devices on the other system, or by copying files to files on the other system and printing them with the /REMOTE qualifier of the PRINT command.

```
$ COPY FY83.DAT EBONY::LPAO:
```

Prints FY83.DAT in your default directory on LPA0 (typically the line printer assigned to SYS\$PRINT) on EBONY. Note that to specify a device (rather than a file) in the COPY command, you must follow the device name with a colon.

```
$ PRINT/REMOTE EBONY::CLERK:[ACCOUNTS]FY83.DAT
```

Prints CLERK:[ACCOUNTS]FY83.DAT on the device assigned to the logical name SYS\$PRINT on the system EBONY. Note that the file must reside on the node on which it is being printed.

You can also use the following commands to affect files on another system: CREATE, DELETE, WRITE (with OPEN and CLOSE), PURGE, and SUBMIT/REMOTE.

### 2.7.3 Access Restrictions

If you receive a protection violation or DECnet-VAX error message in attempting to move a file across systems, you have two choices: (1) if you are writing the file, you can send it to a user account on the other system with MAIL (see Appendix MAIL) or (2) you can follow the node name with an access control string.

An access control string consists of the user name and password of a user account on the system that has access to the directory and/or file you want to access. Enclose the access control string in quotation marks and place it between the node name and the double colon. The following example displays a file located on the system EBONY by including the access control string "PERRY R27L4R15" in the file specification. In this example, PERRY is the user name and R27L4R15 is the password.

```
$ TYPE EBONY"PERRY R27L4R15": :CLERK: [ACCOUNTS]FY83.SUM
```

If the account does not have a password, omit the password from the access control string.

If you define a logical name that includes an access control string, you must use triple quotation marks around the access control string in the definition. The following example equates the logical name CURRENT to a file specification using an access control string (where PERRY is the user name and R27L4R15 is the password).

```
$ DEFINE CURRENT EBONY""PERRY R27L4R15"" : :CLERK: [ACCOUNTS]FY85.SUM
```

## Chapter 3

# Editing Files with the EDT Editor

EDT is an interactive text editor. With EDT you can create a new file, insert text into it, and modify that text. You can also edit text in existing files.

EDT provides both line and keypad editing. In line editing, you type the editing command and the range of text you want the command to affect. In keypad editing, you move the cursor directly to the text you want to change and press keypad keys to enter the editing commands.

An efficient way to use EDT on a video terminal is to use keypad as the primary editing mode in combination with various line-editing commands. You can also redefine certain keypad and control keys to perform editing functions not available in keypad editing. This chapter describes keypad editing as the main editing mode and includes supplementary line-editing commands and key definitions. (If you are using a hardcopy terminal, you can use only line-editing commands. See Appendix EDT.3 for a list of EDT's line-editing commands.)

### 3.1 Invoking and Terminating EDT

The DCL command EDIT invokes the EDT editor, which has its own set of commands for editing files. Included in these EDT commands are two commands for leaving the EDT editor: EXIT and QUIT.

## 3-2 Editing Files with the EDT Editor

### 3.1.1 Invoking EDT

To invoke EDT, type the DCL command EDIT and specify as a parameter the file you want to edit. If the specified file already exists, EDT saves the existing versions and places a copy of the latest version in your buffer. The existing versions of the file remain unchanged. For example, to edit an existing file named MEMO.TXT, enter the following command line:

```
$ EDIT MEMO.TXT
```

```
I am happy to inform you that Justice H. Osgoode has  
accepted our offer of the position of town programmer.  
[EOB]
```

The first few lines of the latest version of the file appear on the screen. The cursor is positioned at the top of the screen, and EDT is ready to receive a keypad-editing command.

If you invoke EDT to create a file, the following message appears:

```
$ EDIT NEWFILE.TXT
```

```
[EOB]
```

**Input file does not exist**

Only the EDT message and the end-of-buffer symbol, [EOB], appear on the screen, and EDT is ready to receive keypad-editing commands. (A buffer is the temporary storage area in which you edit text. See Section 3.6.7 for more information about buffers.)

**NOTE:** In the previous examples, you enter EDT in keypad (change) mode because a startup command file (SYS\$LOGIN:EDTINI.EDT) containing the SET MODE CHANGE command has been executed. If this command is not executed in an EDT startup command file, you will enter EDT in line mode. See Section 3.7.1 for more information on EDT startup command files.

### 3.1.2 Terminating EDT

Both the EXIT and QUIT commands terminate an EDT editing session; however, only EXIT saves your edits in a new version of the file. (Note that the existing versions of a file remains unchanged regardless of how the editing session is terminated.)



### 3.1.2.1 Saving Your Edits

To save your edited text, use the line-editing command EXIT to terminate EDT. When you enter the EXIT command, EDT creates an output file containing the edited version of the input file. By default, the output file will have the same name and type as the input file, with the version number incremented by 1. For example, if you enter the EXIT command after editing a file named MEMO.TXT;3, the output file will be named MEMO.TXT;4.

```
*EXIT
$DISK1: [USER]MEMO.TXT;4  2 lines
$
```

To override the default output file name, enter the EXIT command with a new file specification as the parameter. For example, if you end the same editing session with EXIT OSGOODE.DAT, the output file will be named OSGOODE.DAT;1 (providing no other file named OSGOODE.DAT exists).

```
*EXIT OSGOODE.DAT
$DISK1: [USER]OSGOODE.DAT;1  2 lines
$
```

### 3.1.2.2 Discarding Your Edits

To terminate EDT without saving your edits, use the line-editing command QUIT. All edits you have made to the text will be ignored, and no output file will be created.

```
*QUIT
$
```

The QUIT command is a useful way to terminate EDT when you have opened a file by mistake.

## 3.2 Entering EDT Commands

You enter most keypad-editing commands by pressing a keypad key. You enter line-editing commands by typing them after the line-editing prompt and pressing RETURN.

### 3.2.1 Entering EDT Line Commands

EDT prompts for line-editing commands with an asterisk. Line-editing commands usually operate on a range of one or more lines of text that you specify as a parameter for the command. For example, to display an entire file on your screen, enter the TYPE command and specify WHOLE as the parameter.

```
*TYPE WHOLE
```

You can abbreviate EDT line-editing commands, however, for clarity, the examples in this chapter show complete line-editing commands.

### 3.2.2 Entering Keypad Commands

In keypad editing, the screen displays editing changes as you make them. You type text from the main keyboard and enter keypad-editing commands from the numeric keypad. (To initiate keypad editing you must first enter the line-editing command CHANGE. See Section 3.4.2 for information on the CHANGE command.)

The following figure shows the keypad keys and their functions.

PF1 GOLD	PF2 HELP	PF3 FNDNXT FIND	PF4 DEL L UND L
7 PAGE COMMAND	8 SECT FILL	9 APPEND REPLACE	— DEL W UND W
4 ADVANCE BOTTOM	5 BACKUP TOP	6 CUT PASTE	, DEL C UND C
1 WORD CHNGCASE	2 EOL DEL EOL	3 CHAR SPECINS	ENTER ENTER
0 LINE OPEN LINE	• SELECT RESET	SUBS	

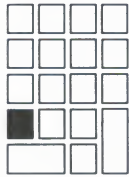
ZK-1688-84

Each key in the keypad performs at least one editing command; most perform two. Pressing a key invokes the regular, or upper, function. To invoke the alternate, or lower, function of a key, press the GOLD key (labeled PF1) first, followed by the desired key. (In the examples that follow, a small diagram of the keypad highlights the key or keys that perform the command being described. The text associated with the keypad illustrates the effect of that editing command.)

For example, key 1 performs both the WORD and the CHNGCASE functions. To invoke the WORD command, press WORD: the cursor moves to the beginning of the next word.

I am happy to inform you that Justice H. Osgoode has  
accepted our offer of the position of town programmer.  
[EOB]

#### WORD

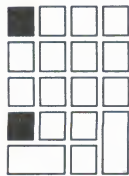


I am happy to inform you that Justice H. Osgoode has  
accepted our offer of the position of town programmer.  
[EOB]

To invoke the CHNGCASE command, press the GOLD key first and then CHNGCASE: the character at the cursor changes from lowercase to uppercase or from uppercase to lowercase.

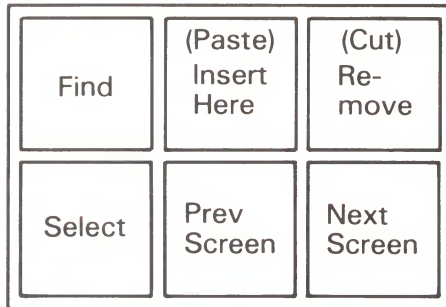
I am happy to inform you that Justice H. Osgoode has  
accepted our offer of the position of town programmer.  
[EOB]

#### CHNGCASE



I Am happy to inform you that Justice H. Osgoode has  
accepted our offer of the position of town programmer.  
[EOB]

The supplemental editing keys on the VT200 keypad perform the same functions as some of the EDT keypad keys. (See Appendix EDT.2.3 for more information about these supplemental editing keys.)



ZK-1677-84

### 3.2.3 Canceling EDT Commands

Use CTRL/C to cancel the currently executing EDT command without affecting previous edits. For example, to stop the display of a long file, press CTRL/C.

\*TYPE WHOLE

.  
.  
.

CTRL/C

Aborted by CTRL/C

\*

The display stops and the CTRL/C message appears.

## 3.3 Getting Help in EDT

EDT provides a help facility for each of the EDT editing modes.

### 3.3.1 Getting HELP on Keypad-Editing Commands

To display a diagram of the keypad keys and their various functions, enter change mode and then press the HELP key (labeled PF2). (On VT200 series terminals, you can also use the HELP key on the supplemental editing keypad.) To display information about a particular keypad command, first press the HELP key and then press the keypad key.

### 3.3.2 Getting HELP on Line-Editing Commands

To display a list of EDT topics on which information is available, type HELP and press RETURN. To display information about a particular command or topic, type HELP followed by the name of the topic and press RETURN. EDT responds with a display of information about the topic and a list of related topics on which information is available. To display information on the use of a particular command qualifier, type HELP plus the command and that qualifier and press RETURN. For example, to display information on the use of /QUERY with the COPY command, enter the following command line:

```
*HELP COPY /QUERY
```

### 3.3.3 Getting HELP on Nokeypad-Editing Commands

Nokeypad commands are used to construct key definitions. To display a list of the nokeypad-editing commands on which information is available, enter the following HELP command in line mode:

```
*HELP CHANGE SUBCOMMANDS
```

To display information about nokeypad entities (units of text upon which nokeypad-editing commands operate), enter the following HELP command in line mode:

```
*HELP CHANGE ENTITIES
```

## 3.4 Changing Editing Modes

You can easily switch back and forth between line and keypad editing; you can also enter line-editing commands from keypad mode. Before using keypad commands, be sure that your terminal type is set properly. (Use SHOW TERMINAL to display the setting and SET TERMINAL/INQUIRE to set the terminal type.)



### 3.4.1 Changing from Keypad to Line Editing

To change from keypad editing to line editing, press CTRL/Z. The asterisk prompt appears at the bottom of your screen, indicating EDT is ready to accept line-editing commands.

I am happy to inform you that Justice H. Osgoode has accepted our offer of the position of town programmer.

[EOB]

CTRL/Z

\*

### 3.4.2 Changing from Line to Keypad Editing

To change from line editing to keypad editing, enter the CHANGE command:

\*CHANGE

The first 22 lines of the file are displayed on your screen. If the file has fewer than 22 lines, the [EOB] symbol will appear below the last line of the file.

### 3.4.3 Entering Line-Editing Commands from Keypad Mode

The keypad COMMAND function allows you to enter line-editing commands without leaving keypad mode. First, enter COMMAND (by pressing GOLD and then COMMAND) to invoke the Command: prompt, then type the line-editing command and press ENTER. (If you press RETURN by mistake, ^M appears; simply delete the ^M by pressing the DELETE key on the main keyboard, and press ENTER.) The following example enters the line-editing command SET QUIET, which suppresses the sound made when EDT issues an error message.

I am happy to inform you that Justice H. Osgoode has accepted our offer of the position of town programmer.

[EOB]

## COMMAND



I am happy to inform you that Justice H. Osgoode has  
accepted our offer of the position of town programmer.  
[EOB]

Command: SET QUIET

ENTER



### 3.5 Recovering from Interruptions

You can recover from interruptions to your editing session in the following ways:

- Deleting extraneous characters—Pressing CTRL/W removes extraneous characters (such as a broadcast message or a message indicating that you have received electronic mail) from the screen and restores the previous display. You should use CTRL/W to clear extraneous characters to ensure that the cursor is in the correct position.
- Resuming an interrupted editing session—The DCL command CONTINUE resumes an editing session that was interrupted by pressing CTRL/Y, so long as only built-in DCL commands were entered after pressing CTRL/Y. (See Section 1.4 for a list of built-in commands.) For example, you could press CTRL/Y, then enter the command SHOW TIME, and then return to your editing session with the CONTINUE command. (Press CTRL/W to restore the screen to its display prior to the SHOW TIME and CONTINUE commands.)
- Recovering a lost session—By default, EDT keeps a journal file with the same file name as the input file and a file type of JOU. If the editing session ends without interruption, the journal file is deleted when you terminate the session. If the editing session is aborted (for example, during a system failure or in response to pressing CTRL/Y, or entering the QUIT/SAVE command), you can recover your edits (with the possible exception of those commands entered just prior to the interruption). Enter the same command line you used to begin the editing session but add the /RECOVER qualifier. For example:

\$ EDIT/RECOVER MEMO.TXT

EDT will reproduce the editing session, reading the commands from the journal file and executing them on the screen.

## 3.6 EDT Keypad Editing

While line editing allows you to manipulate large portions of text easily, EDT keypad editing provides easy manipulation of small units of text. Several EDT keypad commands enable you to find, insert, delete, substitute, and move text in a file. The cursor can be moved through a file in a variety of ways, and the position of the cursor in a file determines how text will be affected by EDT commands.

### 3.6.1 Manipulating the Cursor

You can manipulate the cursor with commands that move it unit by unit through the text and commands that move it directly to a particular location. Several commands that move the cursor are controlled by the ADVANCE and BACKUP commands, which set the cursor's direction forward and backward respectively. Unless otherwise stated, this chapter assumes the default direction of the cursor to be ADVANCE. If your cursor does not move as described, it may be set to move backward rather than forward. See Section 3.6.1.4 for information about changing the cursor's direction. Where EDT editing keys on the VT200 and VT100 series terminals differ, the VT200 key is described first and the VT100 key is described in parentheses.

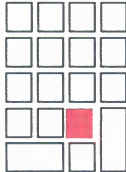
#### 3.6.1.1 Moving the Cursor by Small Units

You can move the cursor by character, word, and line units.

Three keys move the cursor by character.

- RIGHT ARROW—Moves the cursor one character to the right.
- LEFT ARROW—Moves the cursor one character to the left.
- CHAR—Moves the cursor one character in the current direction (depending on whether ADVANCE or BACKUP is set).

```
I am happy to inform you that Justice H. Osgoode has  
accepted our offer of the position of town programmer.  
[EOB]
```

**CHAR**

I am happy to inform you that Justice H. Osgoode has  
accepted our offer of the position of town programmer.  
[EOB]

The WORD command moves the cursor to the beginning of the next or previous word (depending on whether ADVANCE or BACKUP is set).

I am happy to inform you that Justice H. Osgoode has  
accepted our offer of the position of town programmer.  
[EOB]

**WORD**

I am happy to inform you that Justice H. Osgoode has  
accepted our offer of the position of town programmer.  
[EOB]

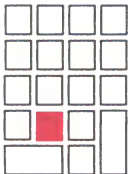
Several keys move the cursor by line:

- UP ARROW—Moves the cursor up one line.
- DOWN ARROW—Moves the cursor down one line.
- EOL—Moves the cursor to the end of the current or previous line (depending on whether ADVANCE or BACKUP is set).

I am happy to inform you that Justice H. Osgoode has  
accepted our offer of the position of town programmer.  
[EOB]

### 3-12 Editing Files with the EDT Editor

#### EOL



I am happy to inform you that Justice H. Osgoode has  
accepted our offer of the position of town programmer.  
[EOB]

- F12 (the BACKSPACE key on VT100 series terminals)—Moves the cursor to the beginning of the previous line.

I am happy to inform you that Justice H. Osgoode has  
accepted our offer of the position of town programmer.

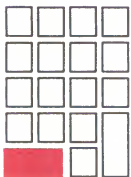
[F12] ( [BACKSPACE] )

I am happy to inform you that Justice H. Osgoode has  
accepted our offer of the position of town programmer.

- LINE—Moves the cursor to the beginning of the next line or previous line, depending upon whether ADVANCE or BACKUP is set.

I am happy to inform you that Justice H. Osgoode has  
accepted our offer of the position of town programmer.

#### LINE



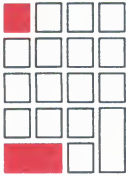
I am happy to inform you that Justice H. Osgoode has  
accepted our offer of the position of town programmer.  
[EOB]

The OPEN LINE command terminates a line without moving the cursor. (The RETURN key also terminates a line, but moves the cursor to the next line.) The OPEN LINE command is useful when you want to insert a blank line or a new line of text. When the cursor is placed at the beginning of a line and the OPEN LINE command is entered, the text on that line is moved down so that the cursor is at the beginning of a blank line.



I am happy to inform you that Justice H. Osgoode has  
accepted our offer of the position of town programmer.  
[EOB]

#### OPEN LINE



I am happy to inform you that Justice H. Osgoode has  
accepted our offer of the position of town programmer.  
[EOB]

#### 3.6.1.2 Moving the Cursor by Large Units

The SECT and PAGE commands allow you to scan several lines of text at a time. The direction in which EDT moves depends upon whether ADVANCE or BACKUP is set.

- SECT—Moves the cursor across a 16-line section of text in EDT's current direction. If there are fewer than 16 lines, SECT moves the cursor across the existing lines.

(On the VT200 series terminals, the supplemental editing keypad key Next Screen moves the cursor 16 lines forward, regardless of EDT's current direction. The supplemental editing keypad key Prev Screen moves the cursor 16 lines backward, regardless of EDT's current direction.)

- PAGE—Moves the cursor to the next or previous page boundary (form feed) or to the end or top of the buffer if there is no boundary. To insert form feeds in your text, use CTRL/L.

#### 3.6.1.3 Moving the Cursor to the Beginning or End of the Buffer

The TOP and BOTTOM commands allow you to move directly to the beginning or end of a buffer. (A buffer is a temporary storage area in which you edit text. See Section 3.6.7 for information about buffers.)

- TOP—Moves the cursor to the beginning, or top, of the buffer.

I am happy to inform you that Justice H. Osgoode has  
accepted our offer of the position of town programmer.  
[EOB]

**TOP**

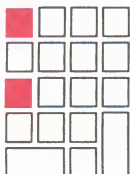


I am happy to inform you that Justice H. Osgoode has  
accepted our offer of the position of town programmer.  
[EOB]

- **BOTTOM**—Moves the cursor to the end, or bottom, of the buffer.

I am happy to inform you that Justice H. Osgoode has  
accepted our offer of the position of town programmer.  
[EOB]

**BOTTOM**



I am happy to inform you that Justice H. Osgoode has  
accepted our offer of the position of town programmer.  
[EOB]

### 3.6.1.4 Changing the Cursor's Direction

The **ADVANCE** and **BACKUP** commands control the cursor's direction for the following EDT keypad-editing commands: **CHAR**, **CHNGCASE**, **EOL**, **FIND**, **FNDNXT**, **LINE**, **PAGE**, **SECT**, **SUBS**, and **WORD**. Each of the directional commands remains in effect until you set the cursor in the opposite direction with the other command.

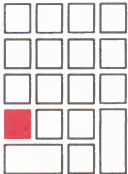
- **ADVANCE**—Sets the cursor's direction forward so that subsequent commands move the cursor in the forward direction. For example, if you enter the **WORD** command after using **ADVANCE**, the cursor moves forward one word.

I am happy to inform you that Justice H. Osgoode has  
accepted our offer of the position of town programmer.  
[EOB]



## 3-16 Editing Files with the EDT Editor

WORD



I am happy to inform you that Justice H. Osgoode has  
accepted our offer of the position of town programmer.  
[EOB]

The cursor will then remain set in the backward direction until you press ADVANCE. For example, if you enter a second WORD command in the example above, you will receive a message indicating that the command requests EDT to back up past the top of the buffer.

The ADVANCE and BACKUP commands are particularly important in string searches; see Section 3.6.4 for more information on searches.

### 3.6.2 Inserting Text

To insert text in EDT keypad editing, position the cursor where you want the text to be inserted and begin typing; the cursor remains one position to the right of the last character inserted. Inserting text in the middle of a line moves both the cursor and the remainder of the line one position to the right for each character inserted. When the line exceeds 80 characters, the text you type will either wrap to the following line or remain invisible, depending on the status of the SET SCREEN, SET [NO]TRUNCATE, and SET [NO]WRAP commands. (See Section 3.7.2 for information about screen formatting commands.)

### 3.6.3 Deleting and Restoring Text

The delete commands work like the cursor movement commands. In EDT keypad editing, you can delete by character using the Delete key (  $\langle X \rangle$  ) (DELETE on VT100 series terminals) and DEL C; by word using F13 (LINEFEED on VT100 series terminals) and DEL W; and by line using DEL L, DEL EOL, and CTRL/U. The deleted text is stored in a buffer so that you can also restore the character (UND C), word (UND W), or line (UND L) most recently deleted wherever and as many times as you desire. Note that the undelete commands restore only the corresponding units of text that were most recently deleted. For example, if you have deleted two lines of text with the DEL L (delete line) command, the UND L (undelete line) command will restore only one line, the line most recently deleted.

### 3.6.3.1 Deleting and Restoring Characters

Whereas the  $\langle X \rangle$  key on the main keyboard (the DELETE key on VT100 series terminals) deletes the character immediately to the left of the cursor, the EDT keypad-editing command DEL C deletes the character directly at the cursor. The UND C command restores the last character deleted with either the  $\langle X \rangle$  (DELETE) key or the DEL C command. For example:

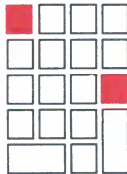
I am happy to inform you that Justice H. Osgoode has  
accepted our offer of the position of town programmer.  
[EOB]

DEL C



am happy to inform you that Justice H. Osgoode has  
accepted our offer of the position of town programmer.  
[EOB]

UND C



I am happy to inform you that Justice H. Osgoode has  
accepted our offer of the position of town programmer.  
[EOB]

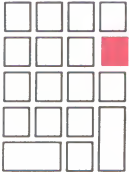
### 3.6.3.2 Deleting and Restoring Words

The F13 key on the main keyboard (the LINEFEED key on VT100 series terminals) deletes to the beginning of the current or preceding word, and the DEL W command deletes to the end of the current word. Blank spaces are considered part of the word they follow, while all other word delimiters are considered to be separate words in themselves. The UND W command restores the last word deleted with either the F13 (LINEFEED) key or the DEL W command. For example:

I am happy to inform you that Justice H. Osgoode has  
accepted our offer of the position of town programmer.  
[EOB]

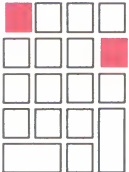


**DEL W**



I am happy to inform you that H. Osgoode has  
accepted our offer of the position of town programmer.  
[EOB]

**UND W**



I am happy to inform you that Justice H. Osgoode has  
accepted our offer of the position of town programmer.  
[EOB]

### 3.6.3.3 Deleting and Restoring Lines

Three commands delete a line (or part of a line) of text.

- **DEL L**—Deletes from the cursor to the end of the line, including the line terminator. If the cursor is at the beginning of the line, the entire line is deleted, and the cursor is positioned at the beginning of the next line.
- **DEL EOL**—Deletes from the cursor to the end of the line (excluding the line terminator), leaving the cursor at the end of the truncated line.
- **CTRL/U**—Deletes from the cursor to the next previous beginning of line, leaving the cursor at the beginning of the previous line. (If CTRL/U is used when the cursor is at the beginning of the line, the previous line is deleted.)

The **UND L** command restores the last line (or part of a line ) that was deleted with the **DEL L**, **DEL EOL**, OR **CTRL/U** command. For example:

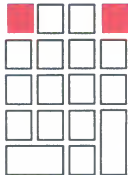
I am happy to inform you that Justice H. Osgoode has  
accepted our offer of the position of town programmer.  
[EOB]

DEL L



I am happy accepted our offer of the position of town programmer.  
[EOB]

UND L



I am happy to inform you that Justice H. Osgoode has  
accepted our offer of the position of town programmer.  
[EOB]

### 3.6.3.4 Deleting and Restoring Large Sections

The EDT line-editing command DELETE is useful for deleting large sections of text. Generally you use line numbers to specify a range for a line-editing command. For example, to delete lines 306 through 860, enter the following:

\*DELETE 306 THRU 860

555 lines deleted

861 Furthermore, I have been informed recently that

Note that the EDT line-editing command SET NUMBERS (the default) must be in effect for line numbers to be displayed in EDT line editing.

You can also use certain keywords (such as WHOLE, REST, BEFORE) as range specifiers. For example, if you are in the middle of a long buffer and want to delete from the cursor to the end of the buffer, enter the following:

Command: DELETE REST

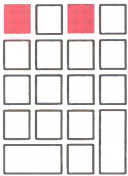
(You can also specify range by using the EDT keypad-editing command SELECT. See Section 3.6.6.1 for information on SELECT.)

### 3.6.4 Locating Text

You can move the cursor to a specified character string with the FIND and FNDNXT EDT keypad-editing commands. The FIND command searches for the specified character string between the current position of the cursor and the beginning or end of the buffer (depending on whether the ADVANCE or the BACKUP command is in control). EDT does not distinguish between uppercase and lowercase letters unless you use the SET SEARCH EXACT line-editing command. When EDT finds the string, it positions the cursor at the first character in the string (unless the SET SEARCH END command is in effect, and then the cursor is positioned at the last character in the string). In a long file the message "Working" may flash on the screen while EDT searches for the string.

For example, to insert a comma after the word "Vermont" in the following text, you can use the FIND command to move the cursor to the string "Vermont." First, enter the EDT keypad command FIND by pressing the GOLD key and then the FIND key (on the VT200 series terminal you may also use the FIND key located on the supplemental editing keypad). Next, type the string you want to locate (the search string) after the Search for: prompt.

#### FIND



I am happy to inform you that Justice H. Osgoode has  
accepted our offer of the position of town programmer.

I am especially glad to have him since he brings not  
only an excellent programming background, but experience  
in town management as well. For those of you who were  
unable to interview him, Mr. Osgoode comes from Steeple  
Creek, Vermont where he served as chief statistician,  
responsible for designing and implementing their current  
budget system. He leaves Steeple Creek in July to join  
us September 1.

[EOB]

Search for: **vermont**

To search in the forward direction, use the ADVANCE command to enter the search string.

# ADVANCE



I am happy to inform you that Justice H. Osgoode has accepted our offer of the position of town programmer.

I am especially glad to have him since he brings not only an excellent programming background, but experience in town management as well. For those of you who were unable to interview him, Mr. Osgoode comes from Steeple Creek, Vermont where he served as chief statistician, responsible for designing and implementing their current budget system. He leaves Steeple Creek in July to join us September 1.

[EOB]

Use the CHAR command to move the cursor to the end of the word "Vermont", and then insert the comma.

# CHAR



I am happy to inform you that Justice H. Osgoode has accepted our offer of the position of town programmer.

I am especially glad to have him since he brings not only an excellent programming background, but experience in town management as well. For those of you who were unable to interview him, Mr. Osgoode comes from Steeple Creek, Vermont, where he served as chief statistician, responsible for designing and implementing their current budget system. He leaves Steeple Creek in July to join us September 1.

[EOB]

## 3-22 Editing Files with the EDT Editor

To find the next occurrence of the string located with the FIND command, use the FNDNXT (find next) command. If there is no other occurrence of the string (as in the example above), EDT issues the message "String was not found."

**NOTE:** Note that the directional setting of the cursor determines the direction of the search. After you press FIND, you can press either ADVANCE or BACKUP (depending on the direction in which you wish to search) to enter the search string. You can also use the ENTER command, which applies the current direction to the search.

### 3.6.5 Substituting Text

To substitute one character string for another, you can use the SUBS keypad-editing command or the SUBSTITUTE line-editing command. The EDT line-editing command can make global substitutions; that is, it can replace every occurrence of one character string in the specified range with another string using only one EDT line-editing command. In contrast, you must use the keypad SUBS command (press the GOLD key followed by the SUBS key) for each substitution you make. (If you do not specify a range, the line-editing command SUBSTITUTE replaces only the first occurrence of the search string in the current line with the substitute string.)

For example, to substitute the string "Osgood" for "Osgoode" throughout a buffer, enter the line-editing command SUBSTITUTE, the old string, and the new string, separating all three with the same delimiter. You can use any nonalphanumeric character (except the percent sign and underscore) as a delimiter for the SUBSTITUTE command, so long as the delimiting character is not part of either string. To apply the command to the entire buffer in a global substitution, specify WHOLE as the parameter. When the operation has been completed, EDT displays each occurrence of the substitution and the total number of substitutions. The following example substitutes the string "Osgood" for each occurrence of the string "Osgoode" in the text below.

#### COMMAND





I am happy to inform you that Justice H. Osgoode has accepted our offer of the position of town programmer.

I am especially glad to have him since he brings not only an excellent programming background, but experience in town management as well. For those of you who were unable to interview him, Mr. Osgoode comes from Steeple Creek, Vermont, where he served as chief statistician, responsible for designing and implementing their current budget system. He leaves Steeple Creek in July to join us September 1.  
[EOB]

Command: SUBSTITUTE\Osgoode\Osgood\WHOLE

ENTER


I am happy to inform you that Justice H. Osgood has accepted our offer of the position of town programmer.

I am especially glad to have him since he brings not only an excellent programming background, but experience in town management as well. For those of you who were unable to interview him, Mr. Osgood comes from Steeple Creek, Vermont, where he served as chief statistician, responsible for designing and implementing their current budget system. He leaves Steeple Creek in July to join us September 1.  
[EOB]

1 I am happy to inform you that Justice H. Osgood has  
7 unable to interview him, Mr. Osgood comes from Steeple  
2 substitutions  
Press return to continue

After you press RETURN, the cursor moves to the top of the buffer.

Note that a global substitution replaces all occurrences of the string, regardless of case or surrounding characters. If you want EDT to search for exact comparisons of case, use the SET SEARCH EXACT command. If the search string occurs in the middle of a longer string, the substitution will still be made. For instance, a global substitution of "IN" for "AT" would change all words containing the string "AT".

## 3-24 Editing Files with the EDT Editor

("LATER" would become "LINER", "THAT" would become "THIN", "SAT" would become "SIN", and so on.)

To get EDT to prompt you before each substitution, use the /QUERY qualifier with the SUBSTITUTE command.

Command: **SUBSTITUTE\AT\IN\WHOLE/QUERY**

EDT prompts you with a ? to verify each substitution. You can respond with one of the following:

- Y Yes, do the substitution
- N No, do not do the substitution
- Q Quit, terminate the command
- A All, do the rest of the substitutions without query

### 3.6.6 Moving Text

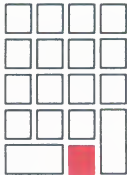
Both EDT keypad and line commands can move text; however, only line-editing commands transfer text between buffers and files.

#### 3.6.6.1 Moving Text Within the File

The EDT keypad-editing command CUT deletes a selected range of text and the PASTE command inserts it at the cursor's current position. (On the VT200 series terminals, the supplemental editing keys Remove and Insert Here perform the same functions as the EDT keypad commands CUT and PASTE.) For instance, to move the first sentence in the second paragraph of the example to the end of that paragraph, move the cursor to the beginning of the sentence and press SELECT. (On the VT200 series terminals, the supplemental editing key SELECT performs the same function as the EDT keypad command SELECT.) This marks the beginning of the selected range. (You can cancel the SELECT command with the RESET command.)

I am happy to inform you that Justice H. Osgood has  
accepted our offer of the position of town programmer.

I am especially glad to have him since he brings not  
only an excellent programming background, but experience  
in town management as well. For those of you who were  
unable to interview him, Mr. Osgood comes from Steeple  
Creek, Vermont, where he served as chief statistician,  
responsible for designing and implementing their current  
budget system. He leaves Steeple Creek in July to join  
us September 1.  
[EOB]

**SELECT**

To mark the end of the selected range, move the cursor to the end of the sentence. The terminal highlights a selected range in reverse video. (The selected range includes the text up to the character preceding the cursor.)

I am happy to inform you that Justice H. Osgood has  
accepted our offer of the position of town programmer.

I am especially glad to have him since he brings not  
only an excellent programming background, but experience  
in town management as well. For those of you who were  
unable to interview him, Mr. Osgood comes from Steeple  
Creek, Vermont, where he served as chief statistician,  
responsible for designing and implementing their current  
budget system. He leaves Steeple Creek in July to join  
us September 1.

[EOB]

Then press CUT to delete the selected text.

**CUT**

I am happy to inform you that Justice H. Osgood has  
accepted our offer of the position of town programmer.

For those of you who were  
unable to interview him, Mr. Osgood comes from Steeple  
Creek, Vermont, where he served as chief statistician,  
responsible for designing and implementing their current  
budget system. He leaves Steeple Creek in July to join  
us September 1.

[EOB]

### 3-26 Editing Files with the EDT Editor

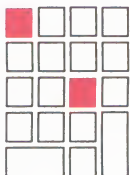
EDT holds the text deleted with the CUT command in the PASTE buffer until you enter the CUT command again. To restore the text, move the cursor to the appropriate position and enter the PASTE command. (The text will be inserted directly in front of the cursor.)

I am happy to inform you that Justice H. Osgood has  
accepted our offer of the position of town programmer.

For those of you who were  
unable to interview him, Mr. Osgood comes from Steeple  
Creek, Vermont, where he served as chief statistician,  
responsible for designing and implementing their current  
budget system. He leaves Steeple Creek in July to join  
us September 1.

[EOB]

#### PASTE



I am happy to inform you that Justice H. Osgood has  
accepted our offer of the position of town programmer.

For those of you who were  
unable to interview him, Mr. Osgood comes from Steeple  
Creek, Vermont, where he served as chief statistician,  
responsible for designing and implementing their current  
budget system. He leaves Steeple Creek in July to join  
us September 1.

I am especially glad to have him since he brings not  
only an excellent programming background, but experience  
in town management as well.

[EOB]

Because the selected text is held in the PASTE buffer until you perform another CUT operation (or give the line-editing command CLEAR PASTE), you can paste the text contained in the PASTE buffer as many times as you want. You can also enter the PASTE buffer to edit the text it contains. (See Section 3.6.7 for information on using multiple buffers.)

After moving the text, you may want to use the FILL command to reorganize selected text so that the maximum number of whole words are fitted within the current line width. The default line width is 80 characters, but you can use the SET WRAP command to use another line length for filling text. For example, you can set the line

length to 60 characters with the EDT line-editing command SET WRAP and then fill a selected range of text.

## COMMAND

I am happy to inform you that Justice H. Osgood has accepted our offer of the position of town programmer.

For those of you who were unable to interview him, Mr. Osgood comes from Steeple Creek, Vermont, where he served as chief statistician, responsible for designing and implementing their current budget system. He leaves Steeple Creek in July to join us September 1.

I am especially glad to have him since he brings not only an excellent programming background, but experience in town management as well.

[EOB]

Command: SET WRAP 60

ENTER


I am happy to inform you that Justice H. Osgood has accepted our offer of the position of town programmer.

For those of you who were unable to interview him, Mr. Osgood comes from Steeple Creek, Vermont, where he served as chief statistician, responsible for designing and implementing their current budget system. He leaves Steeple Creek in July to join us September 1.

I am especially glad to have him since he brings not only an excellent programming background, but experience in town management as well.

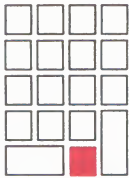
[EOB]



### 3-28 Editing Files with the EDT Editor

EDT will now wrap lines of inserted text and fill lines of selected text at a line width of 60 characters. Use the SELECT command to mark the text you want to affect and then enter the EDT keypad command FILL.

#### SELECT



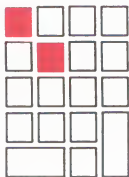
I am happy to inform you that Justice H. Osgood has  
accepted our offer of the position of town programmer.

For those of you who were  
unable to interview him, Mr. Osgood comes from Steeple  
Creek, Vermont, where he served as chief statistician,  
responsible for designing and implementing their current  
budget system. He leaves Steeple Creek in July to join  
us September 1.

I am especially glad to have him since he brings not  
only an excellent programming background, but experience  
in town management as well.

[EOB]

#### FILL



I am happy to inform you that Justice H. Osgood has  
accepted our offer of the position of town programmer.

For those of you who were unable to interview him, Mr.  
Osgood comes from Steeple Creek, Vermont, where he served as  
chief statistician, responsible for designing and  
implementing their current budget system. He leaves Steeple  
Creek in July to join us September 1. I am especially glad  
to have him since he brings not only an excellent  
programming background, but experience in town management as  
well.

[EOB]

### 3.6.6.2 Moving Text Between Files

There are several EDT line-editing commands that move text. For example, the MOVE and COPY commands each perform a function similar to those of the keypad CUT and PASTE operations. MOVE deletes text from one location and inserts it in another; COPY inserts a copy of the text where specified without deleting any text. The EDT line-editing commands INCLUDE and WRITE perform tasks not possible with EDT keypad-editing commands.

- INCLUDE—Copies a file into the buffer you are currently editing or the buffer you specify. Follow the VAX/VMS conventions for file specifications when specifying the file to be copied to the buffer. For example, the following command copies the file named MEM.DAT to the buffer named BUF1.

Command: `INCLUDE MEM.DAT =BUF1`

- WRITE—Copies a specified range of text from a buffer (the current buffer by default) to a specified file. If you do not specify a range, the WRITE command copies the entire contents of the current buffer. For example, the following command copies the contents of the current buffer to the file OSGOODE.DAT.

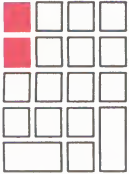
Command: `WRITE OSGOODE.DAT`  
`$DISK1: [USER] OSGOODE.DAT;1 11 lines`

The message displays the new file's specification and length.

### 3.6.7 Using Multiple Buffers

When you begin editing a file with EDT, you are working on a copy of the file in a buffer called MAIN. (EDT also uses a buffer called PASTE to store the text that you delete with the CUT and APPEND commands; you can edit this buffer just as you can edit other text buffers.) You can create other buffers to store pieces of text during your EDT editing session. You can enter and edit these buffers; you can copy text to and from them; and you can write their contents to specified files. To create a buffer, press the COMMAND key. Type the line-editing command FIND followed by the equal sign and whatever name you are giving the buffer, then press the ENTER key. For example, the following command creates a buffer named BUF1.

## COMMAND



I am happy to inform you that Justice H. Osgood has accepted our offer of the position of town programmer.

For those of you who were unable to interview him, Mr. Osgood comes from Steeple Creek, Vermont, where he served as chief statistician, responsible for designing and implementing their current budget system. He leaves Steeple Creek in July to join us September 1. I am especially glad to have him since he brings not only an excellent programming background, but experience in town management as well.

[EOB]

Command: **FIND=BUF1**

When you enter this command, the system responds by displaying nothing but the [EOB] symbol, which indicates that the current buffer, BUF1, is empty. You can now insert and edit text just as you would in the MAIN buffer. To return to the MAIN buffer, follow the same procedure, typing FIND=MAIN rather than FIND=BUF1. To return to your previous position in the MAIN buffer, include a period after the buffer's name.

Command: **FIND=MAIN.**

The buffer named BUF1 will remain intact until you exit from EDT, regardless of whether you issue the EXIT or QUIT command. That is, you can enter, edit, and exit from a buffer at will. However, when you exit from EDT, only the buffer MAIN is saved.

The SHOW BUFFER command displays the number of lines contained in each buffer and indicates (with an equal sign) the current buffer. The following example indicates that there are three buffers (including MAIN and PASTE, which always exist) and that MAIN is the current buffer.



## 3-32 Editing Files with the EDT Editor

I am happy to inform you that Justice H. Osgood has accepted our offer of the position of town programmer.

For those of you who were unable to interview him, Mr. Osgood comes from Steeple Creek, Vermont, where he served as chief statistician, responsible for designing and implementing their current budget system. He leaves Steeple Creek in July to join us September 1. I am especially glad to have him since he brings not only an excellent programming background, but experience in town management as well.  
[EOB]

```
Command:  WRITE BUDGET.DAT =BUF1
```

```
$DISK1: [USER] BUDGET.DAT;1  2 lines
```

EDT returns a message indicating that the file has been created, and the cursor is returned to its previous location in the buffer.

## 3.7 Controlling EDT Sessions

You can control some of the characteristics of an EDT editing session with the SET commands. You can redefine the functions of many keys by using the EDT line-editing command DEFINE KEY (or CTRL/K in keypad editing) plus one or more EDT nokeypad-editing commands. (EDT nokeypad-editing commands perform keypad operations; you can combine them in key definitions to extend your editing capacity. See Appendix EDT.4.2 for a list of EDT nokeypad-editing commands.) You can also define a macro (a sequence of line-editing commands) and define keys in EDT. You can enter these control commands interactively, or you can include them in an EDT startup command file.

### 3.7.1 Startup Command Files

An EDT startup command file contains EDT line-editing commands that are executed when you invoke EDT (before you receive control of the editor). A startup command file is useful for setup operations in EDT; it can include specifications for screen format, definitions of text entities, and definitions of keys and macros. Generally, EDT reads a system-wide startup command file at the beginning of your editing session. If no system-wide startup command file exists on your system, EDT looks for a file named EDTINI.EDT in your default directory and processes the commands in that file.

To create an EDTINI.EDT file, invoke an editor and specify EDTINI.EDT as the file specification. An example follows:

```
$ EDIT EDTINI.EDT
```



Now list the commands, one per line. Some typical commands you might want to put in a startup command file follow:

```
SET QUIET
SET WRAP 60
SET SEARCH BOUNDED
SET TAB 5
DEFINE KEY GOLD P AS "PAR."
SET MODE CHANGE
```

When you exit from the editor, you will have an EDTINI.EDT file. Then, every time you invoke the editor, the commands in your EDTINI.EDT file will be in effect.

To specify an EDT startup command file named something other than EDTINI.EDT, you must include the file specification in the EDIT command line as follows:

```
$ EDIT/COMMAND=startup-file-spec file-spec
```

For a list of EDT line and nokeypad editing commands, see Appendix EDT.4.2.

### 3.7.2 Controlling Screen Format with SET Commands

Several EDT commands control the format of a screen display. A few of these are listed below. See Appendix EDT.3.2 for a comprehensive list of the SET commands.

- SET LINES n—Controls the number of lines that EDT displays on the screen. This number, which can be set from 1 to 22, defaults to 22. To set the screen to 15 lines, for example, type:

```
Command: SET LINES 15
```

Note that if you are editing at slow baud rates, setting the number of lines low will increase your editing speed.

- SET SCREEN width—Controls the maximum length of the line EDT displays; the default width is 80 characters. (When there are more characters than the SET SCREEN command specifies, EDT displays a diamond at the end of the line.)

```
Command: SET SCREEN 132
```

If you SET SCREEN wider than 80 on either a VT100 or VT200 series terminal, EDT changes the terminal's screen width to 132.

- SET [NO]TRUNCATE—Controls whether the characters that exceed the SET SCREEN width are displayed on the next line. The default is SET TRUNCATE, which ends the display of a line at the value of SET SCREEN.

```
Command: SET [NO]TRUNCATE
```

- **SET [NO]WRAP n**—Specifies n character positions as the point at which text will be moved to the beginning of the next line. When you are inserting text in EDT keypad mode and the cursor position exceeds the value of n, EDT wraps the next full word to the next line. (However, when you insert text in the middle of a line, that line does not always wrap.) The default is NOWRAP. To wrap the text exceeding 75 characters, for example, type:

Command: **SET WRAP 75**

The SET commands have corresponding SHOW commands; see Appendix EDT.3.2 for a list of SHOW commands.

### 3.7.3 Controlling Editing Functions with SET Commands

Several commands control EDT's responses during an editing session. Some of these are listed below. (See Appendix EDT.3.2 for a comprehensive list of the SET commands.)

- **SET ENTITY**—Defines boundaries for the WORD, SENTENCE, PARAGRAPH, and PAGE entities. (The SENTENCE and PARAGRAPH entities are not used by any default key definitions; consequently, they are useful only in the key definitions you create with the DEFINE KEY command.) For example, the default boundaries for the WORD entity are a line feed, tab, form feed, line terminator, and space. To make the period and comma the only delimiters of the word entity, give the following SET ENTITY command.

Command: **SET ENTITY WORD ',.'**

- **SET MODE**—Controls the EDT editing mode to be entered when the processing of the EDTINI.EDT file is completed (either line or change mode). For example, to enter change mode instead of line mode at the beginning of editing sessions, insert the following command at the end of your EDT startup command file:  
**SET MODE CHANGE**
- **SET QUIET**—Suppresses the sound made when EDT issues an error message in keypad mode. The default is NOQUIET.

### 3.7.4 Defining Keys

To redefine a key, assign one or more EDT nokeypad-editing commands (listed in Appendix EDT.4.2) to the key with the DEFINE KEY command:

Command: `DEFINE KEY key AS "command(s)"`

You can redefine all keypad keys and you can define any GOLD keyboard key sequence except the keyboard digits, minus sign, and `<X>` key (the DELETE key on VT100 series terminals). See Appendix EDT.3.2 for a diagram of the keypad key numbers you use to define keypad keys. Although you can define many keys as control keys, you should not redefine specialized control keys, such as CTRL/C, CTRL/M, CTRL/O, CTRL/P, CTRL/Q, CTRL/R, CTRL/S, CTRL/T, CTRL/U, CTRL/Y, and CTRL/Z.

The following example of a key definition redefines GOLD + S to perform a global substitution, with prompts for the search string and the replacement string.

`*DEFINE KEY GOLD S AS "EXT S/?*'REPLACE: '/?*' WITH: '/WHOLE."`

The string between quotation marks consists of an EDT nokeypad-editing command (EXT, which tells EDT that the rest of the line is an EDT line-editing command); an EDT line-editing command (SUBSTITUTE, abbreviated "S"); its qualifier (WHOLE); prompts (the question marks followed by the prompts REPLACE and WITH, and the asterisk (\*) directly following the question mark prompt, allowing you to use either ENTER or RETURN to enter your response). Note that the period at the end of the definition (before the final quotation mark) is necessary to make the command execute immediately when the key is pressed. Subsequent use of GOLD + S performs the substitution, prompting for old and new strings and displaying the substitutions.

I am happy to inform you that Justice H. Osgood has  
accepted our offer of the position of town programmer.

For those of you who were unable to interview him, Mr.  
Osgood comes from Steeple Creek, Vermont, where he served as  
chief statistician, responsible for designing and  
implementing their current budget system. He leaves Steeple  
Creek in July to join us September 1. I am especially glad  
to have him since he brings not only an excellent  
programming background, but experience in town management as  
well.

[EOB]

`[GOLD] + [S]`

REPLACE: Osgood `[ENTER]` WITH: Osgoode `[ENTER]`

I am happy to inform you that Justice H. Osgoode has  
unable to interview him, Mr. Osgoode comes from Steeple  
2 substitutions

Placing key definitions (such as the GOLD + S definition) in an EDT startup command file makes the redefined keys available during every editing session.

### 3.7.5 Defining EDT Macros

An EDT macro allows you to execute a sequence of EDT line-editing commands whenever you invoke the macro. To define a macro, follow these steps:

1. Use the EDT line-editing command DEFINE MACRO to define the name of a buffer as the macro name. For example, to define a macro named "heading," type:

Command:    **DEFINE MACRO heading**

2. Create and enter a buffer with the same name as the macro. (See Section 3.6.7 for information about using multiple buffers.)

Command:    **FIND=heading**

3. Type the EDT line-editing commands in the desired sequence, one command per line. The following macro inserts a four-line heading.

```
INSERT;NAME:
INSERT;DEPT:
INSERT;DATE:
INSERT;SUBJ:
[EOB]
```

4. Exit from the buffer.

Command:    **FIND=MAIN.**

To invoke the macro, enter its name as an EDT line-editing command. For example:

Command:    **HEADING**

The lines of the heading are inserted at the cursor position.

```
NAME:
DEPT:
DATE:
SUBJ:
```

To make a macro available during other editing sessions, you can place the DEFINE MACRO command and the macro command sequence in an EDT startup command file. When you include a macro definition in a startup command file, be sure the command sequence contains the commands for entering the macro buffer

(FIND=buffer-name.) and returning to the MAIN buffer (FIND=MAIN.). Note that you must precede each command in the sequence with the INSERT command. For example, to use the macro HEADING in an EDT startup command file precede each insertion with the INSERT command:

```
DEFINE MACRO HEADING
FIND=HEADING
INSERT;INSERT;NAME:
INSERT;INSERT;DEPT:
INSERT;INSERT;DATE:
INSERT;INSERT;SUBJ:
FIND=MAIN
```

You can also redefine a key to accomplish the same results as the macro. The following example defines GOLD + J to insert the heading of the previous macro example:

```
*DEF KEY GOLD J AS "I^Z^MINAME:^Z^MIDEPT:^Z^MIDATE:^Z^MISUBJ:^Z^M."
```

For more information about key definitions and macro definitions, see Appendix EDT.3.





## Chapter 4

### Using DIGITAL Standard Runoff

DIGITAL Standard Runoff (DSR) processes source files into formatted text and intermediate files and creates tables of contents and indexes. You process source and intermediate files with the RUNOFF, RUNOFF/CONTENTS, and RUNOFF/INDEX commands.

The source DSR file is your creation. It can contain text and DSR commands. The DSR commands cause the text to be formatted into sections, paragraphs, lists, and so on. DSR commands start with the control flag, which is usually represented by a period (you can change the character). Five examples of DSR commands follow.

```
.BREAK  
.BLANK  
.BLANK 2  
.RIGHT MARGIN 34  
.CENTER;Twelve Days of Dieting
```

The .BREAK command ends the current line. The .BLANK command without a parameter inserts a blank line in the text. The .BLANK command with the parameter 2 inserts two blank lines in the text. The .RIGHT MARGIN command sets the right margin at position 34. The .CENTER command centers the text following the semicolon.

You can abbreviate DSR command names. The abbreviations must be exactly as listed in Appendix DSR.2. (You cannot always abbreviate simply by shortening the command name to any length as in DCL.) The abbreviations for .BLANK, .CENTER, and .RIGHT MARGIN are .B, .C, and .RM.

DSR commands can be typed in uppercase, lowercase, or a combination of uppercase and lowercase.

On any line containing a DSR command, the first item on the line must be a DSR command and the control flag must occupy position 1. Depending on the particular commands, a line containing a command may contain additional commands and/or text. Note the following examples:

```
.BLANK 2
```

## 4-2 Using DIGITAL Standard Runoff

This DSR command occupies its own line. The end of the line terminates the command.

```
.CENTER;Twelve Days of Dieting
```

This command and text are placed on one line. The semicolon acts as the command terminator.

```
.BLANK 2.CENTER;Twelve Days of Dieting
```

Two commands are placed on one line. The beginning of the second command terminates the first command.

See Appendix DSR.2 for the complete rules for entering DSR commands and for a description of each command. The following example demonstrates the use of the DSR commands .BLANK and .CENTER to format text.

```
.RIGHT MARGIN 34
.CENTER;Twelve Days of Dieting
.CENTER;Watching Your Weight Increase
.BLANK 2
On the twelfth day of dieting, Millitsa gave to me,
.BREAK
Twelve hot fudge sundaes,
.BREAK
Eleven Hostess Twinkies,
.BREAK
Ten cherry cheese cakes,
.BLANK
Nine lady fingers,
.BREAK
Eight date nut muffins,
.BREAK
Seven oatmeal cookies,
.BREAK
Six bags of Fritos,
.BREAK
Five coffee rings,
.BLANK
Four sticky buns,
.BREAK
Three Clark bars,
.BREAK
Two marbled cakes,
.BREAK
And a pizza with pepperoni.
```

The preceding lines coded in DSR produce the following output:

Twelve Days of Dieting  
Watching Your Weight Increase

On the twelfth day of dieting, Millitsa gave to me,  
Twelve hot fudge sundaes,  
Eleven Hostess Twinkies,  
Ten cherry cheese cakes,  
  
Nine lady fingers,  
Eight date nut muffins,  
Seven oatmeal cookies,  
Six bags of Fritos,  
Five coffee rings,  
  
Four sticky buns,  
Three Clark bars,  
Two marbled cakes,  
And a pizza with pepperoni.

You should avoid editing the output file from a DSR operation. You should instead modify the DSR file and reprocess it. If you do make minor modifications to the output file, note that the file does not have the carriage return record attribute (which causes each record in the file to produce a new line automatically when the file is displayed or printed); the carriage return and line feed control characters are embedded in the file.

## 4.1 Formatting Text

You can format text into paragraphs, lists, and literal text. You can adjust the margins and you can adjust the spacing between lines or blocks of lines. DSR also provides commands for leaving blocks of space on a page and for writing notes and footnotes.

You can control such features as bolding and underlining with embedded flags. By default, DSR treats certain characters as flags rather than text. For example, by default, a character is underlined if it is preceded by an ampersand (&). (To place an actual ampersand in the text, you must write an ampersand preceded by an underscore (\_&) or you must turn off the flag governing that character.) One approach is to turn off all flags at the start of your DSR file. The following commands turn off all flags that are on by default.

## 4-4 Using DIGITAL Standard Runoff

Flag Command	Flag Character
.NO FLAGS ACCEPT	Underscore ( _ ) by default
.NO FLAGS COMMENT	Exclamation point ( ! ) by default
.NO FLAGS LOWERCASE	Backslash ( \ ) by default
.NO FLAGS SPACE	Number sign ( # ) by default
.NO FLAGS SUBINDEX	Right angle bracket ( > ) by default
.NO FLAGS UNDERLINE	Ampersand ( & ) by default
.NO FLAGS UPPERCASE	Circumflex ( ^ ) by default

When you need a particular flag, set the flag on (.FLAGS command), write the text that uses the flag, and set the flag off (.NO FLAGS command). (See Section 4.1.9.) You can change the special flag character when you specify the .FLAGS command.

### 4.1.1 Filling and Justifying Text

By default, DSR fills and justifies text. Filling causes words from the input file to be placed on one output line until the addition of another word would exceed the right margin. Justification causes DSR to add spaces between words to expand each line exactly to the right margin. The following example demonstrates how text is filled and justified.

```
.RIGHT MARGIN 45
For it so falls out,
That what we have
we prize not
to the worth
while we enjoy it;
but being lacked,
lacked and lost,
Why,
then we rack the value.
```

The preceding lines coded in DSR produce the following output:

```
For it so falls out,  That  what we have
we prize not to the worth while we enjoy
it; but being lacked, lacked  and  lost,
Why, then we rack the value.
```

If you do not want filling or justification, you must explicitly specify the .NO FILL command or the .NO JUSTIFY command before you write the text. You can turn filling or justifying back on again by specifying the .FILL or .JUSTIFY command. The following example turns off justification but retains filling, which produces a ragged right margin.



```
.RIGHT MARGIN 45
.NO JUSTIFY
For it so falls out,
That what we have
we prize not
to the worth
while we enjoy it;
but being lacked,
lacked and lost,
Why,
then we rack the value.
.JUSTIFY
```

The preceding lines coded in DSR produce the following output:

```
For it so falls out, That what we have
we prize not to the worth while we enjoy
it; but being lacked, lacked and lost,
Why, then we rack the value.
```

The next example turns off both filling and justifying, which formats the output lines in the same way as the input lines.

```
.NO FILL.NO JUSTIFY
For it so falls out,
That what we have
we prize not
to the worth
while we enjoy it;
but being lacked,
lacked and lost,
Why,
then we rack the value.
.FILL.JUSTIFY
```

The preceding lines coded in DSR produce the following output:

```
For it so falls out,
That what we have
we prize not
to the worth
while we enjoy it;
but being lacked,
lacked and lost,
Why,
then we rack the value.
```

The text of the examples in this section is from William Shakespeare's *Much Ado About Nothing*.

### 4.1.2 Adjusting Margins and Centering Text

By default, margins are set at 0 and 70. You can change the margin settings with the .LEFT MARGIN and .RIGHT MARGIN commands. The following example changes the right margin to position 60.

```
.RIGHT MARGIN 60  
<remainder of DSR file>
```

To indent one or more lines of text on the left, increase the left margin setting. After you write the indented text, decrease the left margin setting by the same amount. You can indent on the right by decreasing the right margin setting, writing the text, and increasing the right margin setting. The following example indents text by 10 spaces on either margin.

```
<first part of DSR file>  
.LEFT MARGIN +10.RIGHT MARGIN -10  
<indented text>  
.LEFT MARGIN -10.RIGHT MARGIN +10  
<remainder of DSR file>
```

You can indent a single line of text from the left margin with the .INDENT command. The following example indents a line of text eight spaces.

```
.INDENT 8  
<line of text>
```

You can indent a single line of text from the right margin with the .RIGHT command. You can also use the .RIGHT command to position a single line of text against the right margin.

```
.RIGHT 0  
<line of text>
```

To center text between two margins, use the .CENTER command. You can place the text to be centered on the line following the command, as shown:

```
.CENTER  
<text to be centered>
```

Or you can terminate the .CENTER command with a semicolon and place the text to be centered on the same line.

```
.CENTER;<text to be centered>
```

### 4.1.3 Writing Paragraphs

To separate text into paragraphs, place the .PARAGRAPH command between paragraphs. By default, the .PARAGRAPH command indents the first line of a paragraph five spaces, inserts one blank line before starting a new paragraph, and tests to ensure that room remains on the page for at least four lines of text. You can change the parameter values when you issue your first .PARAGRAPH command or by placing a .SET PARAGRAPH command at the top of the file. The following example does not indent the first line of each paragraph and ensures that room remains on the page for at least three lines of text.

```
.SET PARAGRAPH 0,1,1
<paragraph of text>
.PARAGRAPH
<paragraph of text>
.PARAGRAPH
```

```
.
```

You can also separate text by inserting .BLANK or .SKIP commands. The .BLANK command inserts the exact number of lines specified by the parameter (which defaults to 1). The .BLANK command does not provide for indentation or for testing the room left on the page; you can perform these actions with the .INDENT and .TEST PAGE commands as desired. The following example separates two blocks of text with one blank line, after first testing to ensure that at least three lines remain on the page.

```
<block of text>
.TEST PAGE 3
.BLANK
<block of text>
```

The .SKIP command takes into account the spacing you have in effect. When the default is in effect (spacing is 1), .BLANK and .SKIP are equivalent. However, if multiple spacing is in effect, the .SKIP command multiplies the skip value by the spacing value. You can specify something other than single spacing with the .SPACING command. (The .SPACING command also affects the test page value in the .PARAGRAPH and .SET PARAGRAPH commands.) The following example demonstrates double spacing with two extra lines between blocks of text.

```
.SPACING 2
<block of text>
.SKIP
<block of text>
.SKIP
```

```
.
```

#### 4.1.4 Writing Literal Text

You can place text in the output file exactly as it appears in the DSR file by enclosing it between .LITERAL and .END LITERAL commands.

```
.RIGHT MARGIN 34  
.BLANK 2  
.LITERAL
```

```
    Twelve Days of Dieting  
    Watching Your Weight Increase
```

```
On the twelfth day of dieting, Millitsa gave to me,  
Twelve hot fudge sundaes,  
Eleven Hostess Twinkies,  
Ten cherry cheese cakes,
```

```
Nine lady fingers,  
Eight date nut muffins,  
Seven oatmeal cookies,  
Six bags of Fritos,  
Five coffee rings,
```

```
Four sticky buns,  
Three Clark bars,  
Two marbled cakes,  
And a pizza with pepperoni.  
.END LITERAL
```

The preceding lines coded in DSR produce the following output:

```
    Twelve Days of Dieting  
    Watching Your Weight Increase
```

```
On the twelfth day of dieting, Millitsa gave to me,  
Twelve hot fudge sundaes,  
Eleven Hostess Twinkies,  
Ten cherry cheese cakes,
```

```
Nine lady fingers,  
Eight date nut muffins,  
Seven oatmeal cookies,  
Six bags of Fritos,  
Five coffee rings,
```

```
Four sticky buns,  
Three Clark bars,  
Two marbled cakes,  
And a pizza with pepperoni.
```

Literal text is not filled and not justified—lines are the same as in the input file. Except for the .END LITERAL command, commands and flags are not recognized in literal text. In addition, commands and flags placed before the literal text do not affect the literal text except that you can set the left margin, set tab stops, set spacing, start bolding, and start underlining. You cannot reset any of these items until the literal text ends (for example, you would have to bold the entire literal block). The following example indents the literal text (the .MARGIN commands must be outside the literal block).

```
<text filled and justified>
.LEFT MARGIN +5
.LITERAL
<literal text>
.END LITERAL
.LEFT MARGIN -5
```

If you want the positional effect of literal text but also want to use commands and flags, you can turn off filling and justifying (.NO FILL and .NO JUSTIFY) and turn it back on after the literal text. If the literal text contains blank lines, specify .KEEP when you turn off the filling and justification.

```
<text filled and justified>
.NO FILL
.NO JUSTIFY
.KEEP
<literal text>
.FILL
.JUSTIFY
.NO KEEP
```

For a short number of lines, you might alternatively place .BREAK or .BLANK commands (specify .BLANK 0 for single spacing) between the literal lines.

#### 4.1.5 Writing Lists

You can format text as lists. You can choose among several types of lists; numbered, bulleted, and lettered are possibilities.



#### 4.1.5.1 Numbered Lists

The following three commands format a numbered list:

- `.LIST`—Starts the list by leaving one blank line and indenting. The text of the list is indented nine spaces if the left margin is currently 0. Otherwise, the text of the list is indented four spaces.
- `.LIST ELEMENT`—Starts an element within the list. You can have any number of elements.
- `.END LIST`—Ends the list and writes a blank line.

Each element is preceded by a number starting with number 1. Elements are separated by blank lines.

Note the following example:

```
.LIST  
.LIST ELEMENT;grosbeak  
.LIST ELEMENT;goldfinch  
.LIST ELEMENT;redpoll  
.LIST ELEMENT;sparrow  
.END LIST
```

The preceding lines coded in DSR produce the following output:

1. grosbeak
2. goldfinch
3. redpoll
4. sparrow

The text for each list element can be placed after the semicolon terminating the command (as shown in the example) or can be placed on a line or lines following the command. A single list element continues until the next `.LIST ELEMENT` command or an `.END LIST` command occurs. The list element can contain commands and flags.

You can change the spacing between list elements by specifying the number of spaces as parameter 1 to the `.LIST` command. For example, `.LIST 0` places no spaces between list elements.

#### 4.1.5.2 Bulleted Lists

By default, DSR creates a numbered list. You can substitute another character for the numbers by specifying the character as the second parameter of the .LIST command. The character must be enclosed in quotation marks or apostrophes; the character does not have to be preceded by a comma if the first parameter is not specified. For example, the lowercase o gives the effect of a hollow bullet.

```
.LIST "o"
.LIST ELEMENT;ferret
.LIST ELEMENT;mink
.LIST ELEMENT;rabbit
.LIST ELEMENT;sable
.LIST ELEMENT;raccoon
.END LIST ELEMENT
```

The preceding lines coded in DSR produce the following output:

```
o ferret
o mink
o rabbit
o sable
o raccoon
```

#### 4.1.5.3 Nested Lists

You can nest one list within another as long as the nested list is entirely within one element of the outer list. Note the following example:

```
.LIST O
.LIST ELEMENT;German
.LIST ELEMENT;Russian
.LIST ELEMENT;Swedish
.LIST ELEMENT;Yugoslavian
.LIST O,"o"
.LIST ELEMENT;Serbian
.LIST ELEMENT;Croatian
.LIST ELEMENT;Macedonian
.END LIST O
.LIST ELEMENT;Turkish
.LIST ELEMENT;Scottish
.LIST ELEMENT;Irish
.END LIST
```

The preceding lines coded in DSR produce the following output:

## 4-12 Using DIGITAL Standard Runoff

1. German
2. Russian
3. Swedish
4. Yugoslavian
  - o Serbian
  - o Croatian
  - o Macedonian
5. Turkish
6. Scottish
7. Irish

### 4.1.5.4 Letters and Roman Numerals

By default, DSR numbers lists with decimal numbers. Instead, you can use letters or roman numerals in uppercase or lowercase by specifying them with the `.DISPLAY ELEMENTS` command. You must write the `.DISPLAY ELEMENTS` command after the `.LIST` command for the list and before the first `.LIST ELEMENT` command. The following example writes a numbered list using lowercase roman numerals.

```
.LIST 0
.DISPLAY ELEMENTS RL
.LIST ELEMENT;tan
.LIST ELEMENT;beige
.LIST ELEMENT;rust
.LIST ELEMENT;brown
.END LIST
```

The preceding lines coded in DSR produce the following output:

- i. tan
- ii. beige
- iii. rust
- iv. brown

### 4.1.6 Leaving Space on a Page

The text of a file usually starts on the fourth or fifth line from the top of a page depending on the layout of the document. To leave extra space at the top of a page, specify the amount desired in a `.FIGURE` command (the `.BLANK` command does not work at the top of a page). To leave extra space in the middle of a page, you can use either the `.FIGURE` or `.BLANK` command. If you absolutely must have a certain amount of space all on one page, use the `.FIGURE` or `.FIGURE DEFERRED` command. These commands insert the required space on the next page if it will not fit on the current page. However, the `.FIGURE` command leaves the rest of the current page blank whereas the `.FIGURE DEFERRED` command fills the rest of the current page using the text and commands that follow the `.FIGURE DEFERRED` command.

The following example writes 40 blank lines to the current page if they will fit; otherwise, the 40 blank lines are placed at the top of the next page and the current page is filled from the input lines following the .FIGURE DEFERRED line.

```
<text filled and justified>
.FIGURE DEFERRED 40
<text filled and justified>
```

You can also create space on a page by issuing the .LITERAL command, pressing RETURN once for each empty line, and issuing the .END LITERAL command. This technique allows you to see the amount of space you are creating. However, the space will be split if you cross page boundaries.

#### 4.1.7 Writing Notes

The .NOTE command narrows the left and right margins, inserts a blank line, writes a centered title, inserts a blank line, and writes the text that follows the .NOTE command. The .END NOTE command writes a blank line and restores the margin settings. The title defaults to the word NOTE.

The following example writes a note with the title CAUTION.

```
.NOTE CAUTION
Do not operate the machine outdoors in wet weather.
Do not operate the machine in a wet area indoors or outdoors.
Such actions may lead to a severe electrical shock.
.END NOTE
```

The preceding lines coded in DSR produce the following output:

```
CAUTION

Do not operate the machine outdoors in wet weather.
Do not operate the machine in a wet area indoors or
outdoors. Such actions may lead to a severe
electrical shock.
```

#### 4.1.8 Writing Footnotes

The .FOOTNOTE command places the text following the command at the bottom of the page if enough room exists. If enough room does not exist for the entire footnote, it is placed at the bottom of the next page. No automatic formatting occurs—you must format the footnote with DSR commands as you see fit. In addition, no automatic footnote symbols are provided. The .END FOOTNOTE command ends the footnote and automatically restores any case, fill, justify, and margin settings you might have changed within the footnote.

## 4-14 Using DIGITAL Standard Runoff

The following example demonstrates the use of a footnote.

Press the START button firmly.

Release the START button as soon as the engine starts.(1)

.FOOTNOTE.BLANK.LEFT MARGIN +4.INDENT -4

(1)

If the engine does not crank, ensure that the battery cables are firmly connected to the battery. Sometimes the cables are disconnected for shipping.

.END FOOTNOTE

Push the choke in until you hear it click.

Pull the throttle about halfway down but do not let the engine stall.

After about two minutes, push the choke all the way in and pull the throttle all the way down.

.PARAGRAPH 0

Before engaging the drive train, ensure that you are in a comfortable position to operate the machine.

Two seat controls are provided for your comfort.

The preceding lines coded in DSR produce the following output:

Press the START button firmly. Release the START button as soon as the engine starts.(1) Push the choke in until you hear it click. Pull the throttle about halfway down but do not let the engine stall. After about two minutes, push the choke all the way in and pull the throttle all the way down.

Before engaging the drive train, ensure that you are in a comfortable position to operate the machine. Two seat controls are provided for your comfort.

(1) If the engine does not crank, ensure that the battery cables are firmly connected to the battery. Sometimes the cables are disconnected for shipping.

### 4.1.9 Bolding and Underlining Text

To bold a single character, turn the bold flag on if it is not already on (the bold flag is off by default); in the text, precede the character to be bolded by the bold flag (an asterisk by default); turn the bold flag off to enable the use of the flag as a normal character. The following example bolds the numbers 3 and 7.

.FLAGS BOLD

Follow route \*3 to route \*7.

.NOFLAGS BOLD



To underline a single character, turn the underline flag on if it is not already on (the underline flag is on by default); in the text, precede the character to be underlined by the underline flag (an ampersand by default); turn the underline flag off to enable the use of the flag as a normal character. The following example underlines the letters A and B.

```
.FLAGS UNDERLINE
&A is for Amy and &B is for Basil.
.NOFLAGS UNDERLINE
```

To bold or underline a block of text, turn on the uppercase and lowercase flags if they are not already on (they are on by default) in addition to the bold or underline flag; start the block of text with the uppercase flag (a circumflex by default) followed by the bold or underline flag; end the block of text with the lowercase flag (a backslash by default) followed by the bold or underline flag. The following example bolds a line of text.

```
.FLAGS BOLD
.FLAGS UPPERCASE
.FLAGS LOWERCASE
^*KEEP OFF THE GRASS, PLEASE\*
.NOFLAGS BOLD
.NOFLAGS UPPERCASE
.NOFLAGS LOWERCASE
```

## 4.2 Laying Out a Document

By default (if you do not specify .LAYOUT, .CHAPTER, or .APPENDIX commands), DSR produces a document of consecutively numbered pages. On each page, the text area is the fourth line through the bottom line. Page numbers appear in the upper right corner as "Page 2," "Page 3," and so on, starting with page 2. Running heads (chapter names or other designated text) appear in the upper left corner.

You can adjust the position of page numbers and running heads with the .LAYOUT command. Layout codes 1, 2, and 3 center the page number at the bottom of the page. Layout code 1 centers running heads at the top of the page; layout code 2 moves the running heads to one upper corner or the other depending on whether the page number is odd or even; layout code 3 puts the running heads in the upper left corner and puts the date in the upper right corner. You should specify the .LAYOUT command at the beginning of your file. The following command adjusts the layout to code 2.

```
.LAYOUT 2,3
```

### 4.2.1 Chapters and Appendixes

To break a document into chapters, start each chapter with the `.CHAPTER` command. The title of the chapter must follow the command name on the same line. The lines *following* the `.CHAPTER` command are part of that chapter until another `.CHAPTER` command or an `.APPENDIX` command is given.

By default, chapters are numbered consecutively within the document, beginning with chapter 1. You can force the numbering of a chapter (for example, in order to place each chapter in a separate file) by preceding the `.CHAPTER` command with a `.NUMBER CHAPTER` command. The following example begins chapter 2.

```
.NUMBER CHAPTER 2
.CHAPTER starting procedures
```

The preceding lines coded in DSR produce the following output:

<12 blank lines>

```
CHAPTER 2
STARTING PROCEDURES
```

DSR starts a chapter on a new page with 12 blank lines at the top of the page. The number of the chapter and the chapter title are centered on the page in uppercase. You can adjust the appearance of the chapter number with the `.DISPLAY CHAPTER` command.

The `.APPENDIX`, `.NUMBER APPENDIX`, and `.DISPLAY APPENDIX` commands work similarly to the chapter commands, except that by default appendixes are lettered sequentially starting with appendix A. The following example starts appendix C.

```
.NUMBER APPENDIX C
.APPENDIX connecting the battery
```

### 4.2.2 Sections

The `.HEADER LEVEL` command divides a document into sections and subsections identified by a decimal numbering scheme to a maximum depth of 6. The topmost section is header level 1. Each level is numbered sequentially starting with 1 unless a `.NUMBER LEVEL` command precedes the `.HEADER LEVEL` command. The following example shows a document with three sections.

```
<text>
.HEADER LEVEL 1 normal starting
<text>
.HEADER LEVEL 1 cold weather starting
<text>
.HEADER LEVEL 1 troubleshooting
<text>
```

The preceding lines coded in DSR produce the following output:

```
<text>
1  NORMAL STARTING
<text>
2  COLD WEATHER STARTING
<text>
3  TROUBLESHOOTING
<text>
```

Subsections are numbered within their respective higher-level sections.

```
<text>
.HEADER LEVEL 1 normal starting
<text>
.HEADER LEVEL 2 cold engine
<text>
.HEADER LEVEL 2 warm engine
<text>
.HEADER LEVEL 1 cold weather starting
<text>
.HEADER LEVEL 2 above zero
<text>
.HEADER LEVEL 2 below zero
<text>
.HEADER LEVEL 1 troubleshooting
<text>
```

The preceding lines coded in DSR produce the following output:

```
<text>
1  NORMAL STARTING
<text>
1.1 Cold Engine
<text>
1.2 Warm Engine
<text>
2  COLD WEATHER STARTING
<text>
2.1 Above Zero
<text>
2.2 Below Zero
<text>
3  TROUBLESHOOTING
<text>
```

## 4-18 Using DIGITAL Standard Runoff

If the sections are within chapters or appendixes, the section number is prefixed by the chapter or appendix identifier and a decimal point.

```
.NUMBER CHAPTER 2
.CHAPTER starting procedures
<text>
.HEADER LEVEL 1 normal starting
<text>
.HEADER LEVEL 1 cold weather starting
<text>
.HEADER LEVEL 1 troubleshooting
<text>
```

The preceding lines coded in DSR produce the following output:

```
<chapter heading and text>
2.1 NORMAL STARTING
<text>
2.2 COLD WEATHER STARTING
<text>
2.3 TROUBLESHOOTING
<text>
```

You can force the numbering of a section with the .NUMBER LEVEL command. The following example forces the start of section 1.2.

```
.NUMBER LEVEL 1,2
.HEADER LEVEL 2 Warm Engine
```

You can change the appearance of section headers with the .STYLE HEADERS command. The default .STYLE HEADERS settings cause level 1 headers to be written in all uppercase and level 2 headers to be written with the initial letter of every word in uppercase. The following command changes the settings so that the headers below level 1 are written exactly as you type them.

```
.STYLE HEADERS 3,1,0,7,7,2,1,9,2
```

### 4.2.3 Running Heads

By default, chapter and appendix titles appear as the first line of running heads. If the document does not contain chapters or appendixes, no running heads appear. Running heads are always placed at the top of the page; their exact position can be changed with the .LAYOUT command.

To use level 1 header titles as the second line of running heads, issue the following commands at the start of your DSR file.

```
.SUBTITLE
.AUTOSUBTITLE
```

To use titles other than chapter and section titles as running heads, use the `.TITLE` and `.SUBTITLE` commands. The `.TITLE` command affects the first line of the running head; the `.SUBTITLE` command affects the second line of the running head.

The title specified by a `.TITLE` command remains in effect until another `.TITLE` or `.CHAPTER` command occurs. If you want to use a specified title in place of a chapter name, issue the `.TITLE` command immediately after the `.CHAPTER` command. The title specified by a `.SUBTITLE` command remains in effect until another `.SUBTITLE` command occurs or until a `.HEADER LEVEL` command occurs if automatic subtitles are in effect.

#### 4.2.4 Pagination

If the document is not chapter oriented, pagination is sequential throughout the document. If the document is chapter oriented, pagination is sequential throughout the document only if `.LAYOUT 3` is in effect. Otherwise, pagination is sequential within each chapter and appendix; the page number starts with the chapter or appendix identifier and a hyphen.

You can suspend the numbering of pages with the `.NO NUMBER` command (unless `.LAYOUT 3` is in effect). You can write a document that is not paged (no running heads, no page numbers) by issuing the command `.NO PAGING`.

### 4.3 Processing DSR Files

Enter the `RUNOFF` command to process a DSR file. Specify the name of the DSR file as the parameter; the file type defaults to `RNO`. The following example processes a file named `MESS.RNO` in your default directory.

```
$ RUNOFF MESS
```

If you do not specify the `/OUTPUT` qualifier, the `RUNOFF` command produces an output file with the same name as the input file and a file type of `MEM`. The preceding example produces an output file named `MESS.MEM`. The following example produces an output file named `MESS.MEM` from a DSR file named `TEMPLATE.RNO`.

```
$ RUNOFF/OUTPUT=MESS TEMPLATE
```

See Appendix DSR.1 for a complete description of the `RUNOFF` command and its qualifiers.



### 4.3.1 Producing a Table of Contents

To produce a table of contents, take the following steps:

1. Produce an intermediate (BRN) file—Specify the /INTERMEDIATE qualifier when you process the DSR (RNO) file. The name of the intermediate file is the same as the name of the DSR file but with a file type of BRN, unless you specify a different name. You will also get the usual output (MEM) file; you can specify /NOOUTPUT if you do not want the MEM file.
2. Produce an unformatted table of contents (RNT) file—Issue the command RUNOFF/CONTENTS specifying the intermediate file as input. This command produces a file with the same name as the input file but a file type of RNT.
3. Produce a formatted table of contents (MEC) file—Issue the RUNOFF command specifying the RNT file as input. You must specify the file type. This command produces a file with a file type of MEC containing the formatted table of contents.

The following example processes a file named OPER.RNO and produces an output file named OPER.MEM and a table of contents named OPER.MEC.

```
$ RUNOFF/INTERMEDIATE OPER
$ RUNOFF/CONTENTS OPER
$ RUNOFF OPER.RNT
```

To produce a table of contents from more than one file, you must concatenate the intermediate files when you issue the RUNOFF/CONTENTS command. (You cannot use wildcard characters.) The following example produces output files and a single table of contents from three DSR files.

```
$ RUNOFF/INTERMEDIATE OPER1
$ RUNOFF/INTERMEDIATE OPER2
$ RUNOFF/INTERMEDIATE OPER3
$ RUNOFF/CONTENTS/OUTPUT=OPER OPER1+OPER2+OPER3
$ RUNOFF OPER.RNT
```

The table of contents is based on the .CHAPTER, .APPENDIX, and .HEADER LEVEL commands in your DSR file. You can control the formatting to some extent with the qualifiers to the RUNOFF/CONTENTS command. (See Appendix DSR.1.) You can write additional information to the table of contents with the command .SEND TOC.

### 4.3.2 Producing an Index

An index is based on .INDEX and .ENTRY commands in your DSR file. (You can also use the index flag to index a word of text.) The .INDEX command names an item to be placed in the index. You should position the .INDEX command as close as possible to the text being indexed. The item appears in the index followed by the number of the page on which it was written to the formatted text (MEM) file. The following example makes index entries for each section.

```
<text>
.HEADER LEVEL 1 Normal Starting
.INDEX Normal starting
<text>
.HEADER LEVEL 1 Cold Weather Starting
.INDEX Cold weather starting
<text>
.HEADER LEVEL 1 Troubleshooting
.INDEX Troubleshooting
<text>
```

The index entries would take the following appearance:

```
Cold weather starting, 2-2
Normal starting, 2-1
Troubleshooting, 2-3
```

Use the subindex flag (by default a right angle bracket) to indicate subentries in the index. A subentry is listed under the higher-level item in the index and has its own page number. The subindex flag must be turned on. The following example produces one index entry for "Starting" under which are listed the two subentries "normal" and "cold weather".

```
<text>
.HEADER LEVEL 1 Normal Starting
.FLAGS SUBINDEX
.INDEX Normal starting
.INDEX Starting>normal
.NOFLAGS SUBINDEX
<text>
.HEADER LEVEL 1 Cold Weather Starting
.FLAGS SUBINDEX
.INDEX Cold weather starting
.INDEX Starting>cold weather
.NOFLAGS SUBINDEX
<text>
.HEADER LEVEL 1 Troubleshooting
.INDEX Troubleshooting
<text>
```

## 4-22 Using DIGITAL Standard Runoff

The index entry for "Starting" would have the following appearance:

```
Starting
  cold weather, 2-2
  normal, 2-1
```

You can make an entry without a page number in the index with the .ENTRY command. Usually these index entries are used for cross references. The following example produces an index entry for "Weather" under which the subentry "see cold weather" appears without a page number.

```
.FLAGS SUBINDEX
.ENTRY Weather>see cold weather
```

The index entry would have the following appearance:

```
Weather
  see cold weather
```

To produce an index, take the following steps:

1. Produce an intermediate (BRN) file—Specify the /INTERMEDIATE qualifier when you process the DSR (RNO) file. The name of the intermediate file is the same as the name of the DSR file but with a file type of BRN, unless you specify a different name. You will also get the usual output (MEM) file; you can specify /NOOUTPUT if you do not want the MEM file.
2. Produce an unformatted index (RNX) file—Issue the command RUNOFF/INDEX specifying the intermediate file as input. This command produces a file with the same name as the input file but a file type of RNX.
3. Produce a formatted index (MEX) file—Issue the RUNOFF command specifying the RNX file as input. You must specify the file type. This command produces a file with a file type of MEX containing the formatted index.

The following example processes a file named OPER.RNO and produces an output file named OPER.MEM, a table of contents named OPER.MEC, and an index named OPER.MEX.

```
$ RUNOFF/INTERMEDIATE OPER
$ RUNOFF/CONTENTS OPER
$ RUNOFF/INDEX OPER
$ RUNOFF OPER.RNT
$ RUNOFF OPER.RNX
```

To produce an index from more than one file, you must concatenate the intermediate files when you issue the RUNOFF/INDEX command. (You cannot use wildcard characters.) The following example produces formatted text files, a single table of contents, and a single index from three DSR files.

```

$ RUNOFF/INTERMEDIATE OPER1
$ RUNOFF/INTERMEDIATE OPER2
$ RUNOFF/INTERMEDIATE OPER3
$ RUNOFF/CONTENTS/OUTPUT=OPER OPER1+OPER2+OPER3
$ RUNOFF/INDEX/OUTPUT=OPER OPER1+OPER2+OPER3
$ RUNOFF OPER.RNT
$ RUNOFF OPER.RNX

```

### 4.3.3 Printing Output Files

Keep in mind the following guidelines when printing nonlaser-output files produced by DSR:

- Copying files—You can copy a file (with the COPY command) to a printer, but you may occasionally lose a form feed. More precisely, you lose a form feed when the size of an output page equals 66 or the value that SYS\$LP\_LINES had at the time the file was created if SYS\$LP\_LINES was defined as a logical name. (Note that the current value of SYS\$LP\_LINES has no effect on the output file after it is created.)
- Generating automatic form feeds—In general, you should use the default for the DCL command PRINT (PRINT/FEED) to generate form feeds automatically when printing nonlaser files. However, you may occasionally generate a blank page. More precisely, you generate a blank page when an output page equals 66 or the value that SYS\$LP\_LINES had at the time the file was created if SYS\$LP\_LINES was defined as a logical name. (Note that the current value of SYS\$LP\_LINES has no effect on the output file.) If you specify PRINT /NOFEED, you eliminate the chance of a blank page, but you take the chance of losing pages under the same circumstances as a COPY operation.

You should set up laser printers with the following commands, for example, in the site-specific startup command procedure. *Dsr\$ln01* is the name you assign the laser printer form. *Lpb0* is the name of the printer.

#### 4-24 Using DIGITAL Standard Runoff

```
$ ! Define form for dsr output on ln01 laser printer
$
$ DEFINE/FORM dsr$ln01 /MARGIN=(BOTTOM=0) -
_$ /NOWRAP -
_$ /NOTRUNCATE -
_$ /STOCK=DEFAULT -
_$ /DESCRIPTION="dsr ln01 form definition"
$
$ ! Set up ln01 laser printer for dsr output
$
$ SET PRINTER lpb0 /NOTRUNCATE -
_$ /NOWRAP -
_$ /TAB -
_$ /PRINTALL -
_$ /FF -
_$ /NOCR
```

The print command should specify /NOFEED, the name of the form, and the name of the queue. You should equate this command to a global symbol in your login file or in the system login file. The following example makes LNPRINT a global symbol that prints LN01 files.

```
$ ! Set up lnprint as print command for ln01 laser printer
$
$ LNP*RINT == "PRINT/NOFEED/FORM=dsr$ln01/QUEUE=ln01_queue"
```



## Chapter 5

# Data Representation

At DCL command level, the VAX/VMS operating system permits you to define and run your own images; manipulate character and integer (whole-number) data; create symbols for shorthand purposes or store variable data; and execute commands as batch jobs rather than interactively.

### 5.1 Data Storage

Data is stored as follows:

- Bit—The most basic unit of storage, a bit has a value of 0 or 1.
- Byte—Equal to 8 bits, a byte has an unsigned value of 0 through 255 and a signed value of -128 through 127.
- Word—Equal to 2 bytes (16 bits), a word has an unsigned value of 0 through 65535 and a signed value of -32768 through 32767.
- Longword—Equal to 4 bytes (32 bits), a longword has an unsigned value of 0 through 4294967295 and a signed value of -2147483648 through 2147483647.

A “low-order” unit means the first unit in the series. As a numeric value, the low-order unit is the least significant unit in the number. The following conventions are used in representing the values of storage units:

- Character representation—A series of bytes representing characters is read left to right. The leftmost character represents the low-order byte in the series.
- Binary representation—A series of storage units representing numeric values is read right to left. The rightmost unit represents the low-order unit in the series.

The units in a series of storage units are referenced by their offset from the low-order unit, so that the low-order unit is unit 0, the next unit is unit 1, and so on. Numeric values can be stated in hexadecimal, octal, or decimal radices. The following examples illustrate the conventions for representing data.

## 5-2 Data Representation

**Example 1:** Assume that the word MAINTASK has been stored as a series of characters.

```
Character representation:
    bytes = M A I N T A S K
    offsets = 0 1 2 3 4 5 6 7
Hexadecimal representation on a per-byte basis:
    bytes = 4B 53 41 54 4E 49 41 4D
    offsets = 7 6 5 4 3 2 1 0
Hexadecimal representation on a per-word basis:
    words = 4B53 4154 4E49 414D
    offsets = 3 2 1 0
Hexadecimal representation on a per-longword basis:
    longwords = 4B534154 4E49414D
    offsets = 1 0
Octal representation on a per-byte basis:
    bytes = 113 123 101 124 116 111 101 115
    offsets = 7 6 5 4 3 2 1 0
Decimal representation on a per-byte basis:
    bytes = 75 83 65 84 78 73 65 77
    offsets = 7 6 5 4 3 2 1 0
```

**Example 2:** Assume that the numbers 9001 and 9002 are stored in two consecutive longwords.

```
Decimal representation on a per-longword basis:
    longwords = 9002 9001
    offsets = 1 0
Octal representation on a per-longword basis:
    longwords = 021462 021461
    offsets = 1 0
Hexadecimal representation on a per-longword basis:
    longwords = 00002332 00002331
    offsets = 1 0
Hexadecimal representation on a per-word basis:
    words = 0000 2332 0000 2331
    offsets = 3 2 1 0
Hexadecimal representation on a per-byte basis:
    bytes = 00 00 23 32 00 00 23 31
    offsets = 7 6 5 4 3 2 1 0
Character representation:
    bytes = 1 # . . 2 # . .
    offsets = 0 1 2 3 4 5 6 7
```

(Periods mean the characters are not printable.)

Interpreting characters in octal and decimal on something other than a byte basis makes no sense, as octal and decimal notation do not correspond evenly to the storage capacity of a byte (as does hexadecimal, where one byte represents exactly two digits). Likewise, interpreting numeric values in octal and decimal on something other than the basis of the storage unit representing the number (a longword in the example above) makes no sense.

Right-to-left and offset conventions are also used in referring to bit configurations. For example, the bit configuration of hexadecimal FE (the low-order bit is 0 and the remaining seven bits are 1) is written as 11111110.

### 5.1.1 Character Data

A character is one byte interpreted according to ASCII conventions. Appendix CHAR.2 includes tables of the ASCII character set and the DEC Multinational Character Set. The first half of the numbered columns (0 through 7) in the ASCII table identifies the character as you would enter it on a VT100 or VT200 series terminal or as you would see it on a printer (except for the nonprintable characters). The remaining columns identify the character by the binary value of the byte; the value is stated in three radices—octal, decimal, and hexadecimal. For example, the letter uppercase A has, under ASCII conventions, a storage value of hexadecimal 41 (a bit configuration of 01000001), which is equivalent to 101 in octal notation and 65 in decimal notation.

Characters fall into three main categories:

- **Alphanumeric characters**—The uppercase letters A through Z, the lowercase letters a through z, the digits 1 through 9, the dollar sign (\$), and the underscore (—).
- **Special characters**—All the other characters that can be displayed or printed: the exclamation point (!), quotation marks ("), number sign (#), and so on.
- **Nonprintable characters**—All characters that cannot be printed or displayed. In general, nonprintable characters are ignored for display and print purposes. However, several nonprintable characters serve control functions.

Character	Function
HT	Starts printing or typing at the next horizontal tab
LF	Starts printing or typing on the next line
FF	Starts printing or typing at the top of the next page
CR	Starts printing or typing at the first space on the same line
ESC	Introduces a terminal escape sequence
SP	Inserts one space

## 5-4 Data Representation

EDT and the DUMP command represent the nonprintable characters with special combinations of characters.

Character	Character Name	DUMP	EDT
NUL	Null	^@	^@
SOH-BS	Backspace	^A-^H	^A-^H
HT	Horizontal tab	^I	an actual tab
LF	Linefeed	^J	<LF>
VT	Vertical tab	^K	<VT>
FF	Form feed	^L	<FF>
CR	Carriage Return	^M	<CR>
SO-SUB	Substitute	^N-^Z	^N-^Z
ESC	Escape	^[	<ESC>
FS		^	\\
GS		^]	^]
RS		^^	^^
US		^_	^_
SP		^@	an actual space
DEL		%_	<DEL>

You can specify nonprintable characters with the F\$FAO lexical function and substring assignment statements (see Section 5.5.4). At DCL level, you can generate certain control functions by holding down the CTRL key and typing the letter corresponding to the control function.

CTRL/I	Generates a tab
CTRL/J	Generates a line feed
CTRL/L	Generates a form feed
CTRL/M	Generates a carriage return

A character string is a series of characters interpreted as a single entity. At DCL command level and within command procedures, you specify character strings in either of two formats:

- Names—In name format, you enter the characters that constitute the character string. The name can only include alphanumeric characters and some special characters, depending on the use of the character string. Lowercase characters are converted to uppercase. You use name format mainly for specifying command names, qualifier names, some parameter and qualifier values, and symbol names.

\$ MOUNT/SYSTEM DUA1 ACCOUNTS ACCOUNTS

MOUNT, SYSTEM, DUA1, ACCOUNTS, and ACCOUNTS are character strings specified in name format.

- **Literals**—In literal format, you enclose the characters that constitute the character string in quotation marks. The literal can include all printable characters. (To include a quotation mark in a character literal, type two consecutive quotation marks.) Lowercase characters remain lowercase characters. You use literal format mainly for assigning character strings to symbols and writing records with the WRITE command.

```
$ DOG2 = "No tag, light brown, 30 lbs."
$ OPEN/WRITE POUND POUND.DAT
$ WRITE POUND "No tag, light brown, 30 lbs."
$ CLOSE POUND
```

Do not continue a literal from one line to the next. Assign the literal as several literals to different symbols and concatenate the symbols.

```
$ DOG2A = "No tag, light brown, 30 lbs."
$ DOG2B = "- looks part beagle"
$ DOG2 = DOG2A + DOG2B
```

### 5.1.2 Numeric Data

A number is a series of decimal digits (the ASCII characters 0 through 9), hexadecimal digits (the ASCII characters 0 through 9 and A through F), or octal digits (the ASCII characters 0 through 7) interpreted as an integer. The number must be in the range -2147483648 through 2147483647. (An error is not reported if a number outside this range is specified or calculated, but an incorrect value results.)

At DCL command level and within command procedures, you specify a number as follows:

- **Positive numbers**—Specify a positive number by typing the appropriate digits. The following example assigns the number 13 to the symbol DOG\_COUNT.

```
$ DOG_COUNT = 13
$ SHOW SYMBOL DOG_COUNT
DOG_COUNT = 13    Hex = 0000000D    Octal = 000015
```

- **Negative numbers**—Precede a negative number with a minus sign.

```
$ BALANCE = -15237
$ SHOW SYMBOL BALANCE
BALANCE = -15237    Hex = FFFFC47B    Octal = 142173
```

- **Radix**—Specify a number in a radix other than decimal by preceding the number (but not the minus sign) with %X for hexadecimal numbers and %O for octal numbers.



## 5-6 Data Representation

```
$ DOG_COUNT = %XD
$ SHOW SYMBOL DOG_COUNT
DOG_COUNT = 13    Hex = 0000000D    Octal = 000015

$ BALANCE = -%X3B85
$ SHOW SYMBOL BALANCE
BALANCE = -15237    Hex = FFFFC47B    Octal = 142173
```

- Fractions—A number cannot include a decimal point. In calculations, fractions are truncated; for example, 5 divided by 2 equals 2.

Numbers are stored internally as signed 4-byte integers: positive numbers have values of 0 through 2147483647 and negative numbers have values of 4294967296 minus the absolute value of the number. The number -15237, for example, is stored as 4294952059. Negative numbers are converted back to minus-sign format for ASCII or decimal displays; however, they are not converted back for hexadecimal and octal displays. For example, the number -15237 appears in displays as hexadecimal FFFFC47B (decimal 4294952059) rather than hexadecimal -00003B85.

Numbers are stored in text files as a series of digits using ASCII conventions (the digit 1 has a storage value of 49, and so on).

### 5.1.3 Logical Data

Some operations interpret numbers and character strings as logical data with values as follows:

- True—A number has a logical value of true if it is odd (that is, the low-order bit is 1). A character string has a logical value of true if the first character is an uppercase or lowercase T or Y.
- False—A number has a logical value of false if it is even (that is, the low-order bit is 0). A character string has a logical value of false if the first character is not an uppercase or lowercase T or Y.

In both of the following examples, DOG\_COUNT is assigned the value 13 (IF STATUS means if the logical value of STATUS is true).

#### Example 1:

```
$ STATUS = 1
$ IF STATUS THEN DOG_COUNT = 13
```

#### Example 2:

```
$ STATUS = "TRUE"
$ IF STATUS THEN DOG_COUNT = 13
```

## 5.2 Expressions

Expressions are formed by combining data entities with operators. Operators are denoted by:

- Special characters—Asterisk (\*), slash (/), plus sign (+), and minus sign (-).
- Special names—.EQ., .GE., .GT., .LE., .LT., .NE., .NOT., .AND., and .OR.; the names can be in uppercase or lowercase.

Data entities and operators can be adjacent or can be separated by any number of spaces or tabs. The data entities can be symbols or literals. Expressions take two forms:

- Operations—An operation combines two data entities or alters a data entity. The following example combines the values 10 and 3 by adding them.

```
$ DOG_COUNT = 10 + 3
$ SHOW SYMBOL DOG_COUNT
DOG_COUNT = 13   Hex = 0000000D   Octal = 000015
```

- Comparisons—A comparison evaluates a relationship between two entities as true or false. A true comparison evaluates to a numeric value of 1, and a false comparison evaluates to a numeric value of 0. The following example compares the value of DOG\_COUNT with 13 and finds them to be equal.

```
$ DOG_CHECK = DOG_COUNT .EQ. 13
$ SHOW SYMBOL DOG_CHECK
DOG_CHECK = 1   Hex = 00000001   Octal = 000001
```

### 5.2.1 Character Expressions

In a character expression, the data entities must be literal character strings or symbols equated to character strings. Attempting an operation or comparison between a character string and a number causes the character string to be converted to a number.

You can specify the following character operations:

- Concatenation—The plus sign concatenates two character strings.

```
$ COLOR = "light brown"
$ WEIGHT = "30 lbs."
$ DOG2 = "No tag, " + COLOR + ", " + WEIGHT
$ SHOW SYMBOL DOG2
DOG2 = "No tag, light brown, 30 lbs."
```

## 5-8 Data Representation

- Reduction—The minus sign removes the character string in the second specified data entity from the character string in the first specified entity.

```
$ SHOW SYMBOL DOG2
DOG2 = "No tag, light brown, 30 lbs."
$ DOG2 = DOG2 - ", 30 lbs."
$ SHOW SYMBOL DOG2
DOG2 = "No tag, light brown"
```

If the character string in the second specified entity occurs more than once in the first specified entity, only the first occurrence of the string is removed.

You can specify the types of character comparisons given in the following list. When you make such comparisons, strings are compared character by character, and strings of different lengths are not equal (for example, "dog" is greater than "dog" because DCL pads the shorter string with NUL characters). The comparison criteria are the ASCII values of the characters, so that the digits 0 through 9 are less than the letters A through Z, and the uppercase letters A through Z are less than the lowercase letters a through z. A character string comparison terminates either (1) when all the characters have been compared, in which case the strings are equal, or (2) when the first mismatch occurs. For the following examples, assume that LAST\_NAME has the value WHITFIELD.

- Equal to—The operator .EQS. compares one character string to another for equality. The following example indicates that the value of the symbol LAST\_NAME does not equal the literal NORMAN.

```
$ TEST_NAME = LAST_NAME .EQS. "NORMAN"
$ SHOW SYMBOL TEST_NAME
TEST_NAME = 0    Hex = 00000000    Octal = 000000
```

- Greater than or equal to—The operator .GES. compares one character string to another for a greater or equal value in the first specified string. The following example indicates that the value of the symbol LAST\_NAME is greater than or equal to the literal NORMAN.

```
$ TEST_NAME = LAST_NAME .GES. "NORMAN"
$ SHOW SYMBOL TEST_NAME
TEST_NAME = 1    Hex = 00000001    Octal = 000001
```

- Greater than—The operator .GTS. compares one character string to another for a greater value in the first specified string. The following example indicates that the value of the symbol LAST\_NAME is greater than the literal NORMAN.

```
$ TEST_NAME = LAST_NAME .GTS. "NORMAN"
$ SHOW SYMBOL TEST_NAME
TEST_NAME = 1    Hex = 00000001    Octal = 000001
```

- Less than or equal to—The operator .LES. compares one character string to another for a lesser or equal value in the first specified string. The following example indicates that the value of the symbol LAST\_NAME is not less than or equal to the literal NORMAN.

```
$ TEST_NAME = LAST_NAME .LES. "NORMAN"
$ SHOW SYMBOL TEST_NAME
TEST_NAME = 0    Hex = 00000000    Octal = 000000
```

- Less than—The operator .LTS. compares one character string to another for a lesser value in the first specified string. The following example indicates that the value of the symbol LAST\_NAME is not less than the literal NORMAN.

```
$ TEST_NAME = LAST_NAME .LTS. "NORMAN"
$ SHOW SYMBOL TEST_NAME
TEST_NAME = 0    Hex = 00000000    Octal = 000000
```

- Not equal—The operator .NES. compares one character string to another for inequality. The following example indicates that the value of the symbol LAST\_NAME does not equal the literal NORMAN.

```
$ TEST_NAME = LAST_NAME .NES. "NORMAN"
$ SHOW SYMBOL TEST_NAME
TEST_NAME = 1    Hex = 00000001    Octal = 000001
```

## 5.2.2 Numeric Expressions

In a numeric expression, the data entities must be literal numbers or symbols equated to numbers. In addition, you can use a character string that represents a number (for example, "23" or "-51"). Attempting an operation or comparison between a number and a character string causes the character string to be converted to a number.

You can specify the following numeric operations:

- Multiplication—The asterisk multiplies two numbers.

```
$ BALANCE = 142 * 14
$ SHOW SYMBOL BALANCE
BALANCE = 1988    Hex = 000007C4    Octal = 003704
```

- Division—The slash divides the first specified number by the second specified number.

```
$ BALANCE = BALANCE / 14
$ SHOW SYMBOL BALANCE
BALANCE = 142    Hex = 0000008E    Octal = 000216
```

If a number does not divide evenly, the remainder is lost. (No rounding takes place.)

## 5-10 Data Representation

- Addition—The plus sign adds two numbers.  
\$ BALANCE = BALANCE + 37  
\$ SHOW SYMBOL BALANCE  
BALANCE = 179 Hex = 000000B3 Octal = 000263
- Subtraction—The minus sign subtracts the second specified number from the first specified number.  
\$ BALANCE = BALANCE - 15416  
\$ SHOW SYMBOL BALANCE  
BALANCE = -15237 Hex = FFFFC47B Octal = 142173
- Unary plus and minus—The plus and minus signs change the sign of the number they precede.  
\$ BALANCE = -(-142)  
\$ SHOW SYMBOL BALANCE  
BALANCE = 142 Hex = 0000008E Octal = 000216

You can specify the following numeric comparisons:

- Equal to—The operator .EQ. compares one number to another for equality. The following example indicates that BALANCE equals -15237.  
\$ TEST\_BALANCE = BALANCE .EQ. -15237  
\$ SHOW SYMBOL TEST\_BALANCE  
TEST\_BALANCE = 1 Hex = 00000001 Octal = 000001
- Greater than or equal to—The operator .GE. compares one number to another for a greater or equal value in the first number. The following example indicates that BALANCE is greater than or equal to -15237.  
\$ TEST\_BALANCE = BALANCE .GE. -15237  
\$ SHOW SYMBOL TEST\_BALANCE  
TEST\_BALANCE = 1 Hex = 00000001 Octal = 000001
- Greater than—The operator .GT. compares one number to another for a greater value in the first number. The following example indicates that BALANCE is not greater than -15237.  
\$ TEST\_BALANCE = BALANCE .GT. -15237  
\$ SHOW SYMBOL TEST\_BALANCE  
TEST\_BALANCE = 0 Hex = 00000000 Octal = 000000
- Less than or equal to—The operator .LE. compares one number to another for a lesser or equal value in the first number. The following example indicates that BALANCE is less than or equal to -15237.  
\$ TEST\_BALANCE = BALANCE .LE. -15237  
\$ SHOW SYMBOL TEST\_BALANCE  
TEST\_BALANCE = 1 Hex = 00000001 Octal = 000001



- Less than—The operator `.LT.` compares one number to another for a lesser value in the first number. The following example indicates that `BALANCE` is not less than `-15237`.

```
$ TEST_BALANCE = BALANCE .LT. -15237
$ SHOW SYMBOL TEST_BALANCE
TEST_BALANCE = 0    Hex = 00000000    Octal = 000000
```

- Not equal to—The operator `.NE.` compares one number to another for inequality. The following example indicates that `BALANCE` equals `-15237`.

```
$ TEST_BALANCE = BALANCE .NE. -15237
$ SHOW SYMBOL TEST_BALANCE
TEST_BALANCE = 0    Hex = 00000000    Octal = 000000
```

### 5.2.3 Logical Expressions

A logical operation affects all the bits in the number being acted upon. (If the logical data item is a character string beginning with `T`, `t`, `Y`, or `y`, it is treated as the number 1. If the logical data item is a character string beginning with any character other than `T`, `t`, `Y`, or `y`, it is treated as the number 0.) Typically, however, you are interested only in the value of the low-order bit, that is, whether the entity has a logical value of true or false. You can specify the following logical operations:

- Not—The operator `.NOT.` reverses the bit configuration of a logical value. A true value becomes false and a false value becomes true. The following example reverses a true value.

```
$ SHOW SYMBOL STATUS
STATUS = 1    Hex = 00000001    Octal = 000001
$ STATUS = .NOT. STATUS
$ SHOW SYMBOL STATUS
STATUS = -2    Hex = FFFFFFFF    Octal = 177776
```

The value `-2` is even, that is, false.

- And—The operator `.AND.` combines two logical values as follows:

Bit Level	Entity Level
1 .AND. 1 = 1	true .AND. true = true
1 .AND. 0 = 0	true .AND. false = false
0 .AND. 1 = 0	false .AND. true = false
0 .AND. 0 = 0	false .AND. false = false

## 5-12 Data Representation

The following example combines a true value and a false value to produce a false value.

```
$ STAT1 = "TRUE"
$ STAT2 = "FALSE"
$ STATUS = STAT1 .AND. STAT2
$ SHOW SYMBOL STATUS
STATUS = 0    Hex = 00000000    Octal = 000000
```

- Or—The operator .OR. combines two logical values as follows:

Bit Level	Entity Level
1 .OR. 1 = 1	true .OR. true = true
1 .OR. 0 = 1	true .OR. false = true
0 .OR. 1 = 1	false .OR. true = true
0 .OR. 0 = 0	false .OR. false = false

The following example combines a true value and a false value to produce a true value.

```
$ STAT1 = "TRUE"
$ STAT2 = "FALSE"
$ STATUS = STAT1 .OR. STAT2
$ SHOW SYMBOL STATUS
STATUS = 1    Hex = 00000001    Octal = 000001
```

### 5.2.4 Combined Operations and Precedence

An expression can contain any number of operations and comparisons. You can indicate precedence by placing operations to be performed first in parentheses. (Parentheses can be nested.) Otherwise, operations within an expression are evaluated in the following order:

1. Unary plus (+) and minus (−)
2. Multiplication and division
3. All other numeric and character operations
4. All numeric and character comparisons
5. Logical NOT operations
6. Logical AND operations
7. Logical OR operations

Operations and comparisons that have the same precedence are evaluated from left to right. The following examples illustrate precedence of operations in expressions.

**Example 1:**

```
$ BALANCE = 150 + 20 * 4
      BALANCE = 150 + 80
$ SHOW SYMBOL BALANCE
BALANCE = 230   Hex = 000000E6   Octal = 000346
```

**Example 2:**

```
$ BALANCE = (150 + 20) * 4
      (BALANCE = 170 * 4)
$ SHOW SYMBOL BALANCE
BALANCE = 680   Hex = 000002A8   Octal = 001250
```

**Example 3:**

```
$ STATUS = 150 * 4 .GT. 80 * 2
      STATUS = 600 .GT. 160
$ SHOW SYMBOL STATUS
STATUS = 1   Hex = 00000001   Octal = 000001
```

## 5.3 Lexical Functions

A lexical function performs operations on system and user data items, and substitutes the result of the operation for itself. Appendix LEX describes each lexical function and lists them by function.

You invoke a lexical function by typing its name (which always begins with F\$) and a parameter list in place of a character string or number. Use the following format:

**F\$function (parameter,...)**

The parameter list follows the function name with none or any number of intervening spaces and tabs. The list must be enclosed in parentheses. Within the list, specify parameters in exact order and separate them with commas; even if you omit an optional parameter, do not omit the comma. If no parameters are required, type an empty set of parentheses.

The F\$LOCATE function, for example, requires two character strings as parameters and returns (that is, substitutes for itself) a number that is the starting offset of the first character string (the substring) within the second. In the following example, F\$LOCATE is equivalent to the number 4.

**F\$LOCATE ("string","fullstring")**

## 5-14 Data Representation

You can use a lexical function in any position that you can use a symbol. In positions where symbol substitution must be forced by enclosing the symbol in apostrophes, lexical function evaluation must be forced by placing the lexical function within apostrophes. Lexical functions can also be used as parameter values in other lexical functions. Consider the following examples:

```
$ L = F$LENGTH (LINE)
```

Equates the length of the character symbol LINE to a numeric symbol named L.

```
$ LINE = F$EXTRACT (0,F$LENGTH(LINE)-2,LINE)
```

Strips the last two characters from the character string that is the value of the symbol LINE.

## 5.4 Date and Time

You specify time in absolute or delta format or in a combination of both formats. Absolute format provides an exact time. Delta format provides an offset from the current date and time. An absolute and delta time combination provides an exact time plus or minus an offset from the current date and time.

### 5.4.1 Absolute Time

The general format for an absolute time is as follows:

**dd-mmm-yyyy:hh:mm:ss.cc**

where:

dd	day of the month; an integer in the range 1-31
mmm	month; JAN, FEB, MAR, APR, MAY, JUN, JUL, AUG, SEP, OCT, NOV, or DEC
yyyy	year; an integer
hh	hour; an integer in the range 0-23
mm	minute; an integer in the range 0-59
ss	seconds; an integer in the range 0-59
cc	hundredths of a second; an integer in the range 0-99

You can truncate the date or the time on the right. However, if you are specifying both date and time, you must include the colon between the date and time, and the date must contain at least one hyphen. You can omit fields if you supply the punctuation. Truncated or omitted date fields default to the corresponding fields for the current date. Truncated or omitted time fields default to 0.

You can also specify an absolute time as TODAY (00:00:00.00 today), TOMORROW (00:00:00.00 tomorrow), or YESTERDAY (00:00:00.00 yesterday).

If you specify a past time in a command that expects the current or a future time, the current time is used.

Specification	Time
15-MAY-1986:13	1 P.M. on May 15, 1986
15-MAY	0 A.M. on May 15 this year
15:30	3:30 P.M. today
15-	0 A.M. on the 15th of this month
15-::30	0:30 A.M. on the 15th of this month

## 5.4.2 Delta Time

The general format of a delta time is as follows:

dddd-hh:mm:ss.cc

where:

dddd number of days; an integer in the range 0-9999  
 hh number of hours; an integer in the range 0-23  
 mm number of minutes; an integer in the range 0-59  
 ss number of seconds; an integer in the range 0-59  
 cc number of hundredths of seconds; an integer in the range 0-99

You can truncate a delta time on the right. You can omit fields as long as you include the punctuation. Truncated or omitted fields default to 0.

Specification	Time
3-	3 days from now
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

## 5.4.3 Absolute and Delta Time Combinations

To combine absolute and delta time, specify an absolute time plus (+) or minus (-) a delta time. The variable fields and default fields for absolute and delta time values are the same as those described in the preceding sections. The delta time value must always be preceded by a plus or minus sign. Whenever a plus sign precedes the delta time value, the entire time specification must be enclosed in quotation marks.



Specification	Time
" +5"	5 hours from now
" +:5"	5 minutes from now
-:5	current time minus 5 minutes
"TOMORROW+1"	tomorrow plus 1 hour

**NOTE:** If a qualifier is described as a value that may be expressed as an absolute time, a delta time, or a combination of the two, then you must specify a delta time as if it were part of a combination time. For example, to specify a delta time value of five minutes from the current time, use "+:5" (not "0-0:5").

## 5.5 Symbols

A symbol is a name that represents a numeric, character, or logical value. The symbol name is 1 through 255 characters long and must begin with a letter, an underscore (\_), or a dollar sign (\$). In a symbol name, both lowercase and uppercase letters are treated as uppercase.

A symbol can be either of the following:

- **Local**—A local symbol can be accessed from the command level that defines it or from lower command levels.
- **Global**—A global symbol can be accessed from any command level regardless of the level at which it was defined.

Local symbols take precedence over global symbols of the same name. Symbols take precedence over identical command names. If you define a symbol with the same name as a DCL command, your definition overrides the command name. For example, if you define HELP as the command TYPE HELP.LST you cannot use the system's HELP command by typing HELP.

The SHOW SYMBOL command displays symbol values. Specify the name of the symbol to display the value of a particular local symbol. Specify the name of the symbol and /GLOBAL to display the value of a particular global symbol. Specify /ALL to display all local symbols, and /ALL /GLOBAL to display all global symbols.

### 5.5.1 Creation and Deletion

Symbols are created when assigned a value in the following format:

symbol-name [=] value

To create a local symbol, use a single equal sign in the assignment statement; to create a global symbol, use two equal signs. The following commands define the local symbol FILE as the character string ACCOUNTS:[BOLIVAR]PRICES.84 and the global symbol MAX\_VALUE as the number 24.

```
$ FILE = "ACCOUNTS:[BOLIVAR]PRICES.84"
$ MAX_VALUE == 24
```

You can omit the quotation marks around character strings in assignment statements if you precede the equal sign(s) with a colon. Note that symbol assignments that omit quotation marks automatically change the character string to uppercase letters and compress multiple spaces and/or tabs to a single space. The following example creates the same local symbol FILE that the preceding example creates.

```
$ FILE := ACCOUNTS:[BOLIVAR]PRICES.84
```

You can equate symbols to character strings (A="ar"), integers (VAR=26), symbols (C=A), lexical functions (I = F\$LOCATE("substring", "fullstring") ), or a combination of these entities.

The result of DCL's evaluation of a symbol is either a character string or an integer value. The data type (character or integer) of a symbol is determined by the data type of the value currently assigned. The type is not permanent: if the value changes type, as in the following example, the symbol changes type. In this example, the local symbol NUM is first assigned a character value and then converted to an integer value when used in an expression with an integer.

```
$ NUM = "12"
$ RESULT = NUM + 12
```

The DELETE/SYMBOL command deletes a symbol. You must include the /GLOBAL qualifier to delete a global symbol. In the following example, the global symbol TEMP is deleted.

```
$ DELETE/SYMBOL/GLOBAL TEMP
```

### 5.5.2 Symbol Access Control

The SET SYMBOL/SCOPE=(keyword,...) command controls access to local and global symbols in command procedures. This allows you to treat symbols as being undefined without deleting them. Symbol scoping works differently for local and global symbols.

If you specify /SCOPE=NOLOCAL, all local symbols defined in an outer procedure level are treated as being undefined by the current procedure and any inner levels. Specifying LOCAL removes any symbol translation limit set by the current procedure level.

For example, if SET SYMBOL/SCOPE=NOLOCAL was specified at procedure levels 2 and 4, procedure level 2 can access only procedure level 2 local symbols. Procedure level 3 can access procedure levels 2 and 3 local symbols and procedure level 4 can access procedure level 4 local symbols and any local symbols in inner procedure levels.

Global symbols are not procedure level dependent. The global symbol scoping state (GLOBAL or NOGLOBAL) that is in effect when a new procedure level is invoked is propagated to the new procedure level. Specifying /SCOPE=NOGLOBAL makes all global symbols inaccessible for all subsequent commands until you either specify /SCOPE=GLOBAL or exit to a previous level at which global symbols were accessible.

In the following example, the SET SYMBOL command denies access to all global symbols.

```
$ SET SYMBOL/SCOPE=NOGLOBAL
```

Exiting a procedure level back to an outer procedure level causes the symbol scoping state to be restored for both local and global symbols.

### 5.5.3 Symbol Substitution

When a command line is executed, symbols in the following positions are automatically substituted.

- On the right side of an = or == assignment statement
- In a lexical function
- In the brackets on the left side of an assignment statement when you are performing substring substitution (see Section 5.5.4).
- In a DEPOSIT, EXAMINE, IF, or WRITE command
- At the beginning of the command line

To force substitution of a symbol that is not in one of the positions listed, enclose the symbol with apostrophes. For example:

```
$ TYPE 'B'
```

To force substitution of a symbol within a quoted character string, enclose that symbol in double apostrophes, as follows:

```
$ T = "TYPE ''B''"
```

When processing a command line, DCL replaces symbols with their values in the following order:

- Forced substitution—From left to right, DCL replaces all strings delimited by apostrophes (or double apostrophes for strings within quotation marks). Symbols preceded by single apostrophes are translated iteratively; symbols preceded by double apostrophes are not.
- Automatic substitution—From left to right, DCL evaluates each value in the command line, executing it if it is a command and evaluating it if it is an expression. Symbols in expressions are replaced by their assigned values; this substitution is not iterative.

The following example demonstrates the effect of the order in which DCL substitutes symbols. Assume the following symbol definitions:

```
$ SUBN = "SUBMIT/NOTIFY/NOPRINT"
$ FILE1 = "[BOLIVAR]ACCOUNTING.COM"
$ NUM = 1
```

Given these symbol definitions, the following commands submit the command procedure named [BOLIVAR]ACCOUNTING.COM.

```
$ FILE = "'FILE' 'NUM' '"
$ SUBN 'FILE'
```

In the first command, forced substitution causes NUM to become 1, making FILE'NUM' become FILE1. Then automatic substitution causes FILE1 to become [BOLIVAR]ACCOUNTING.COM. In the second command, forced substitution causes 'FILE' to become [BOLIVAR]ACCOUNTING.COM, and then automatic substitution causes SUBN to become SUBMIT/NOTIFY/NOPRINT.

### 5.5.4 Substring Substitution

The following variations of the assignment statement allow you to manipulate character symbols on a character-by-character basis and numeric symbols on a bit-by-bit basis.

- **Character**—Specify a character symbol on a character-by-character basis as follows. The assignment statement must use `:=` in place of `=` to force equation of a character string to a substring.

`symbol[offset,length] := value`

*Offset* specifies the starting character within the symbol. An offset of 0 means the first character in the symbol, an offset of 1 means the second character, and so on. *Length* specifies the number of characters to change. The following example changes the string "THIS IS A STRING" to "THERE'S A STRING".

```
$ STR1 = "THIS IS A STRING"
$ STR1[0,6] := "THERE'"
```

If the specified length is greater than the number of characters in the value, the value is padded on the right with blanks. The following command defines the symbol `LINE` as a character string of 80 blanks.

```
$ LINE[0,80] := " "
```

- **Number**—Specify a number symbol on a bit-by-bit basis as follows:

`symbol[offset,length] = value`

*Offset* specifies the starting bit within the number, where 0 means the low-order bit, 1 means the second bit, and so on. *Length* specifies the number of bits to change. The maximum for both length and offset is 32 bits. (If you have not predefined the symbol that you are planning to set bits in, DCL assumes that the value of that symbol is 0.) The following example defines the symbol `BELL` as the value 7 (7 is the ASCII value of CTRL/G, which rings the terminal's bell).

```
BELL[0,8] = 7      ! the binary value of 7 is 111
```

The low-order byte of `BELL` has the binary value 00000111. By changing the 0 at offset 5 of `BELL` (beginning with 0, count bits from right to left) to 1, you create the binary value 00100111 (hexadecimal 27, the ASCII value for an apostrophe).

```
BELL[5,1] = 1      ! the binary value of 1 is 1
```



### 5.5.5 Use of Symbols

Although you can use symbols as a mnemonic device (for example, equate a phone number to a name: HELEN = 617-754-8076), usually you use symbols as shorthand. You define a symbol as a literal and then type the symbol name rather than the literal. For example, you may define a symbol as any of the following:

- Foreign command—Defining a symbol as a foreign command allows you to execute the named image by entering only the symbol name. In the following example, the symbol FIX is defined to execute the image NUMFIX.EXE in the [BILLS] directory on the disk ACCOUNTS.

```
$ FIX == "$ACCOUNTS:[BILLS]NUMFIX"
```

- Command line—Defining a symbol as a command line allows you to execute the command by entering only the symbol name. In the following example, the symbol EBONY is defined to establish a network connection to the node EBONY.

```
$ EBONY == "SET HOST EBONY"
```

Setting a symbol equal to a command line that executes a command procedure allows you to execute the procedure by typing only the symbol name. In the following example, COUNT is defined to execute the command procedure CENSUS.

```
$ COUNT == "@CENSUS"
```

When you enter COUNT to execute CENSUS, place any parameters for CENSUS after the symbol as if you had entered @CENSUS.

- Character string—Defining a symbol as a character string allows you to insert that string in a command line by typing the symbol (with delimiting apostrophes to force substitution). In the following example, the symbol FILE is first defined as a complete file specification and then used in the TYPE command.

```
$ FILE == "CLERK:[LICENSES.DOG]DOGS81.DAT"
```

```
$ TYPE 'FILE'
```

The string could be a directory you often access. In the following example, whenever the symbol DOG occurs in a command line, the literal value replaces the symbol before the line is executed.

```
$ DOG == "[LICENSES.DOG]"
```

```
$ COPY 'DOG' DOGS83.DAT DOGS83.TMP
```

Symbols can also hold variables: values that you calculate, or that you assign as something other than a literal. For example, you might assign the value of a lexical function to a variable or read the value of a file record into a variable. As variables, symbols are most often used in command procedures (see Chapter 6). Note that the symbols you create are deleted when you log out.



## Chapter 6

### Command Procedures

Command procedures are files containing DCL commands that are executed when the procedure is invoked. A command procedure can be:

- Simple—The procedure executes a series of DCL commands exactly as they are written. The following example sets your default directory and examines it.

```
$ ! PROCEDURES.COM
$ !
$ ! Enter [MAINT.PROCEDURES] and examine it
$ SET DEFAULT [MAINT.PROCEDURES]
$ DIRECTORY
```

- Complex—The procedure performs programlike functions. The following example asks for directory names and examines the directories.

```
$ ! DIRECTORY.COM
$ !
$ ! Examine directories
$START:
$ INQUIRE DIR_NAME "Directory name"
$ IF DIR_NAME .EQS. "" THEN GOTO END
$ DIRECTORY 'DIR_NAME'
$ GOTO START
$END:
```

## 6-2 Command Procedures

### 6.1 Format

Begin each command line with a dollar sign. Omit the dollar sign for data lines. The following format conventions are recommended:

- Use comments—Comments explain the procedure to anyone who must maintain it; they are ignored during execution of the command procedure. Use comments at the beginning of a procedure to describe the procedure and the parameters passed to it; use them at the beginning of each block of commands to describe that section of the procedure. Comments must begin with an exclamation point; the comment is the text to the right of the exclamation point. (To include a literal exclamation point in a command line, precede and follow it with quotation marks.)
- Do not abbreviate—Commands and qualifiers are usually self-explanatory if they are spelled out. Also, when new commands are added with a release of the VAX/VMS operating system, abbreviated commands may no longer be unique and thus command procedures using them might require changes.
- Put labels on separate lines—Using separate lines for labels makes loops and conditional coding easier to understand. Placing labels immediately after the dollar sign (in contrast to commands, which are placed after a blank space following the dollar sign) makes them easier to find. A label can have up to 255 characters, cannot contain embedded blanks, and must be terminated by a colon.
- Separate command sequences—Insert lines containing a dollar sign and an exclamation mark before and after a logical sequence of commands. This makes it easier to see the outline of the command procedure. (If you insert blank lines, DCL reads them as data lines and produces a message warning you that the data lines were ignored.)

### 6.2 Execution

To execute command procedures, type an at sign (@) followed by the file specification of the procedure. The file type defaults to COM. Procedures can be executed at the DCL level or from within another command procedure. The following command executes the procedure SETD.COM in the directory [MAINT.PROCEDURES] on the disk WORKDISK.

```
$ @WORKDISK: [MAINT.PROCEDURES] SETD
```

To simplify the invocation of a procedure, create either a symbol or logical name. Equating the command line to a symbol allows you to invoke the command procedure from any directory by entering the symbol name.

```
$ SETD = "@WORKDISK: [MAINT.PROCEDURES] SETD"  
$ SETD
```

Equating the file specification to a logical name allows you to invoke the command procedure from any directory by entering an at sign followed by the logical name.

```
$ DEFINE SETD WORKDISK: [MAINT.PROCEDURES]SETD.COM.  
$ @SETD
```

### 6.2.1 Nesting Command Levels

Command procedures can contain other nested command procedures. When a command procedure is invoked, the command level increases by 1. For instance, if you invoke procedure SUB from DCL command level (level 0), SUB executes at command level 1. If SUB then calls SUB1, which calls SUBSUB1, SUB1 executes at command level 2 and SUBSUB1 at command level 3. Command levels are limited to 32.

By convention, DCL is called the highest command level and command level 32 the lowest command level. Hence, if you move from command level 3 to command level 2, you are said to be moving to the next higher command level.

### 6.2.2 Exiting from Command Procedures

A command procedure exits when it reaches the end of the procedure, an EXIT command, or a STOP command. If the exit is caused by the end of the procedure or an EXIT command, control returns to the next higher command level. For instance, if you invoke SUB at DCL command level, and SUB calls SUB1:

- Exiting from SUB1 returns you to SUB at the command line following the call to SUB1.
- Exiting from SUB returns you to DCL command level.

If the exit is caused by the STOP command, control returns to DCL command level. If the exit is caused by the EXIT command, you can return a status value to the next higher command level by specifying the value as the parameter of the EXIT command. This status value is placed in the global symbol \$STATUS (see Section 1.9).



## 6.3 Passing Data

Command procedures frequently require data provided by a user. To specify the same data each time the command procedure is executed, place the data on data lines following the command that requires the data. (A data line is a line that does not begin with a dollar sign. To include a data line that begins with a dollar sign, use the DECK and EOD commands.) The following command procedure executes the image CENSUS.EXE, which reads 1981, 1982, and 1983 each time the procedure is executed.

```
$ RUN CENSUS
1981
1982
1983
```

The text on a data line is passed directly to the image; it is not processed by DCL. Therefore, you should not include symbols or arithmetic expressions on data lines. Since logical names are not translated by DCL, a logical name included on a data line is properly processed.

To specify different data each time a command procedure is executed, use one of the following mechanisms:

- Pass the data as one or more parameter values.
- Use the INQUIRE or READ command within the command procedure to read the data.
- Specify a device or file from which to read the data by redefining the logical name SYS\$INPUT.

### 6.3.1 Using Parameters To Pass Data

When you invoke a command procedure, you can pass it up to eight parameters by placing the values of the parameters after the file specification of the command procedure. Separate the parameters with one or more spaces and/or tabs. Specify a parameter value as one of the following:

- **Number**—To pass a number, specify the number. A number must be an integer and is passed as a literal. In the following example, the values "24" and "25" are passed to ADDER.COM.  
\$ @ADDER 24 25
- **Literal**—To pass a literal, specify the literal. In the following example, the values "Paul" and "Cramer" are passed to DATA.COM.  
\$ @DATA Paul Cramer

To preserve spaces, tabs, or lowercase characters, you must place quotation marks before and after the literal. In the following example, the single value "Paul Cramer" is passed to DATA.COM.

```
$ @DATA "Paul Cramer"
```

- Symbol—To pass the value of a symbol, place quotation marks before and after the symbol. In the following example, the values "Paul" and "Cramer" are passed to DATA.COM.

```
$ NAME = "Paul Cramer"
```

```
$ @DATA 'NAME'
```

Note that in passing a symbol, DCL removes quotation marks that enclose a literal. To preserve spaces, tabs, and lowercase characters in a symbol value, you must enter the enclosing quotation marks as part of the symbol value. To include a quotation mark as part of a literal value, enter three quotation marks. In the following example, the single value "Paul Cramer" is passed to DATA.COM.

```
$ NEW_NAME = "" "Paul Cramer" ""
```

```
$ @DATA 'NEW_NAME'
```

- Null string—To pass a null parameter, use a set of quotation marks as a place holder in the command string. In the following example, the first parameter passed to DATA.COM is a null parameter.

```
$ @DATA "" "Paul Cramer"
```

Parameters are treated as literals and placed in the local symbols P1 through P8; P1 is assigned the first parameter value; P2 the second; P3 the third, and so on. If you pass more than eight values, you receive the error message "too many command procedure parameters - limit to eight" and the procedure is not executed; if you pass fewer than eight values, the extra symbols are assigned null values.

For example, when DATA.COM is invoked with the command:

```
$ @DATA "Paul Cramer" 24 "(603) 423--8769"
```

P1 through P8 are defined in DATA.COM as

```
P1 = Paul Cramer
```

```
P2 = 24
```

```
P3 = (603) 423-8769
```

```
P4-P8 = Null
```

When you enter a nested procedure, the local symbols P1 through P8 are assigned the parameters passed by the invoking procedure. The local symbols P1 through P8 in the nested procedure are not related to the local symbols P1 through P8 in the invoking procedure. In the following example, DATA.COM invokes NAME.COM with the command.

```
$ ! DATA.COM
```

```
$ @NAME 'P1'
```

## 6-6 Command Procedures

In NAME.COM, P1 through P8 are defined as follows:

```
P1 = Paul
P2 = Cramer
P3-P8 = Null
```

Because DCL removes quotation marks when passing a symbol, to preserve spaces, tabs, and lowercase characters in a symbol value, you must enclose the value in three sets of quotation marks. In the following example, the literal value in P1 is enclosed in three sets of quotation marks and passed to NAME.COM. If P1 originally contained the value "Paul Cramer", the value "Paul Cramer" is passed to NAME.COM.

```
$ ! DATA.COM
$ QUOTE = ""
$ P1 = QUOTE + P1 + QUOTE
$ @NAME 'P1'
```

An alternative is to enclose the text in quotation marks and, where a symbol appears, precede it with two apostrophes, and follow it with one apostrophe.

```
$ ! DATA.COM
$ @NAME '' 'P1' ''
```

### 6.3.2 The INQUIRE Command

The INQUIRE command prompts for a value, reads the value from the terminal, and assigns it to a symbol. All characters typed at the terminal in response to the prompt are taken as a character string value. By default the response is converted to uppercase, multiple blanks and tabs are replaced by a single space, and leading and trailing spaces are removed. To preserve lowercase characters, multiple spaces, and tabs, enclose your response in quotation marks. The following command procedure writes the prompt Filename: and puts the response into the local symbol FILE.

```
$ INQUIRE FILE "Filename"
```

To suppress the colon and space automatically added to the end of the prompt, use the /NOPUNCTUATION qualifier. To make the symbol global instead of local, use the /GLOBAL qualifier. The following command procedure writes the prompt Do you want to use defaults? and puts the response into the global symbol DEFAULT.

```
$ INQUIRE/NOPUNCTUATION/GLOBAL DEFAULT-
"Do you want to use defaults?"
```

When a command procedure is submitted as a batch job, the value for a symbol specified in an INQUIRE command is read from the data line following the INQUIRE command. If you do not include a data line, the symbol is assigned a null value.

### 6.3.3 The READ Command

The READ command prompts for a value, reads the value from the source specified by the first parameter, and assigns it to the symbol named as the second parameter. By default, the READ command uses the prompt DATA:. To specify a different prompt, use the /PROMPT qualifier. All characters typed on the terminal in response to the prompt are taken as an exact character string value (case, spaces, and tabs are preserved). The following command procedure writes the prompt Filename:, reads the response from the source specified by the logical name SYS\$COMMAND (by default, the terminal), and assigns the response to the symbol FILE.

```
$ READ/PROMPT="Filename:" SYS$COMMAND FILE
```

### 6.3.4 Obtaining Data from SYS\$INPUT

Commands, utilities, and other system images normally take their input from the source specified by the logical name SYS\$INPUT. You can specify SYS\$INPUT as any one of the following:

- **Data line**—In a command procedure, SYS\$INPUT by default is equated to the data lines of the procedure. In the following command procedure, the image CENSUS.EXE uses the default value of SYS\$INPUT to take input (1981, 1982, and 1983) from the data lines.

```
$ ! CENSUS.COM
$ !
$ ! Execute CENSUS
$ RUN CENSUS
1981
1982
1983
```

- **Terminal**—A command procedure can get input from a terminal by defining SYS\$INPUT as the terminal. This allows you to perform interactive tasks from a command procedure. The following command procedure defines SYS\$INPUT as the terminal, then invokes EDIT. The editing session that follows the call to EDIT is interactive. (The /USER\_MODE qualifier redefines SYS\$INPUT for a single image; you should use this qualifier whenever you redefine a process-permanent logical name.)

```
$ ! EDIT.COM
$ !
$ ! Edit the file STATS.DAT
$ WRITE SYS$OUTPUT "Edit STATS.DAT:"
$ DEFINE/USER_MODE SYS$INPUT SYS$COMMAND:
$ EDIT STATS.DAT
```

## 6-8 Command Procedures

- **File**—A command procedure can get input from a file by defining SYS\$INPUT as a file. The following command procedure defines SYS\$INPUT as the file YEARS.DAT, then invokes the program CENSUS. CENSUS reads its input from the file YEARS.DAT. (The /USER\_MODE qualifier redefines SYS\$INPUT for a single image; you should use this qualifier whenever you redefine a process-permanent logical name.)

```
$ ! CENSUS.COM
$ !
$ ! Execute CENSUS
$ DEFINE/USER_MODE SYS$INPUT YEARS.DAT
$ RUN CENSUS
```

## 6.4 Returning Data

To return a value from a command procedure (either to a calling procedure or to DCL command level), you must assign the value to a global symbol. The global symbol can be read at any command level. Use comments to explain the use of any global symbols.

To create a global symbol, specify the value to be passed on the right side of a global assignment statement. In the following example, the command procedure DATA.COM invokes the command procedure NAME.COM, passing NAME.COM a full name. NAME.COM places the last name in the global symbol LAST\_NAME. When NAME.COM completes, DCL continues executing DATA.COM, which reads the last name by referencing the global symbol LAST\_NAME. (The command procedure NAME.COM would be in a separate file; it is indented here for clarity.)

```
$ @DATA "Paul Cramer"

$ ! DATA.COM
$ !
$ ! P1 is a full name
$ ! NAME.COM returns the last name in the
$ ! global symbol LAST_NAME
$
$ @NAME 'P1'
    $ ! NAME.COM
    $ ! P1 is a first name
    $ ! P2 is a last name
    $ ! return P2 in the global symbol LAST_NAME
    $ LAST_NAME := 'P2'
    $ EXIT
$ ! write LAST_NAME to the terminal
$ WRITE SYS$OUTPUT "LAST_NAME = ''LAST_NAME'"
LAST_NAME = "CRAMER"
```



## 6.5 Displaying Data

Commands, utilities, and other system images normally write their output to the source specified by the logical name SYS\$OUTPUT. By default, SYS\$OUTPUT is equated to the terminal. However, you can redirect the output of a command procedure to a file with the /OUTPUT qualifier. In the following example, output from the command procedure SETD.COM is written to the file RESULTS.TXT instead of to the terminal.

```
$ @SETD/OUTPUT=RESULTS.TXT
```

Other DCL commands that accept the /OUTPUT qualifier include: ACCOUNTING, CALL, DIRECTORY, HELP, LIBRARY, RUN (process), SPAWN, and TYPE.

### 6.5.1 Displaying Literals and Symbols

The WRITE command displays literals and symbols on the terminal:

- **Literal**—Enclose the text to be displayed in quotation marks. The following example displays the text "Two files were written."  
\$ WRITE SYS\$OUTPUT "Two files were written."
- **Symbol value**—Enclose the symbol in apostrophes. The following example displays the text "STAT1.DAT".  
\$ FILE = "STAT1.DAT"  
\$ WRITE SYS\$OUTPUT 'FILE'
- **Combination of literals and symbol values**—Enclose the text to be displayed in quotation marks. Preface a symbol with two apostrophes and follow it with one apostrophe. The following example displays the text "STAT1.DAT and STAT2.DAT were written."  
\$ AFILE = "STAT1.DAT"  
\$ BFILE = "STAT2.DAT"  
\$ WRITE SYS\$OUTPUT "'AFILE' and ''BFILE' were written."

### 6.5.2 Displaying Text

To display text that is more than one line long, use the TYPE command. TYPE writes data to SYS\$OUTPUT (the terminal, by default). Using SYS\$INPUT as the parameter causes TYPE to read the data from the command procedure. When the following command procedure is executed, the text on the data lines is displayed on the terminal.

```
$ ! CLEAN.COM
$ !
$ TYPE SYS$INPUT
$ INQUIRE COMMAND-
  "Command (EXIT, DIRECTORY, TYPE, PURGE, DELETE, COPY)"
  .
  .
  .
```

### 6.5.3 Displaying Files

To display a file, use the TYPE command. The following example displays the file STAT1.DAT on the terminal.

```
$ TYPE STAT1.DAT
```

## 6.6 Inputting and Outputting Data from Files (File I/O)

To move data to and from files, use the OPEN, CLOSE, READ, and WRITE commands. The logical name you specify in the OPEN command is used to refer to the file in the WRITE, READ, and CLOSE commands.

### 6.6.1 Writing to a File

To write data to a file, take the following steps:

1. Open the file—The OPEN command opens a file and associates the file name with a logical name.

Use the /APPEND qualifier of the OPEN command to write data to the end of an existing file. If you use the /APPEND qualifier to open a nonexistent file, an error occurs.

Use the /WRITE qualifier of the OPEN command to create a new file. If you use the /WRITE qualifier to open an existing file, a new version of that file is created.

Both the /APPEND and the /WRITE qualifiers cause DCL to open the file for write access.

2. Begin the write loop—File I/O is done in a loop unless you are writing or reading a single record.
3. Read the data to be written—Use the INQUIRE command or the READ command to read data into a symbol.
4. Test the data—Check the symbol containing the data. If the symbol is null (the user pressed RETURN without any data on the line), you have reached the end of the data to be written to the file and should go to the end of the loop. Otherwise, continue.
5. Write the data to the file—Use the WRITE command to write the value of the symbol (one record) to the file.
6. Return to the beginning of the loop—You remain in the loop until there is no more data to be written to the file.
7. End the loop and close the file—The CLOSE command disassociates the file name from the logical name and closes the file. (Files opened by the OPEN command remain open until you log out unless you explicitly close them.)

The following command procedure writes data to the new file STAT.DAT. If a file of that name exists, a new version is created.

\$ ! Write a file	
\$ OPEN/WRITE IN_FILE STAT.DAT	!Open the file
\$ ON CONTROL_Y GOTO END_WRITE	!Close the file if you abort
\$	! execution with a CTRL/Y
\$WRITE:	!Begin loop
\$ INQUIRE STUFF "Input data"	!Get input
\$ IF STUFF .EQS. "" THEN GOTO END_WRITE	!Test for end of file
\$ WRITE IN_FILE STUFF	!Write to the file
\$ GOTO WRITE	!Goto beginning
\$END_WRITE:	!End loop
\$	
\$ CLOSE IN_FILE	!Close the file

**NOTE:** The logical name in the OPEN command must be unique. If the OPEN command does not work and your commands seem correct, change the logical name in the OPEN command. The SHOW LOGICAL command displays logical name definitions.

If you want to create a file with a unique name, use the F\$SEARCH lexical function (see Appendix LEX) to see whether the name is already in the directory. The following command procedure prompts the user for a file name, then uses the F\$SEARCH lexical function to search the default directory for the name. If a file with that name already exists, control is passed to ERROR\_1, the procedure prints the message "File already exists," and control returns to GET\_NAME where the user is again prompted for a file name.

## 6-12 Command Procedures

```
$ ! FILES.COM
$ !
$GET_NAME:
$ ! Get a file name
$ INQUIRE FILE "File"
$ ! Make sure the name is unique
$ CHECK = F$SEARCH (FILE)
$ IF CHECK .NES. "" THEN GOTO ERROR_1
$ ! Open and write to the file
$ OPEN/WRITE IN_FILE 'FILE'
.
.
.
$ EXIT
$ERROR_1:
$ WRITE SYS$OUTPUT "File already exists"
$ GOTO GET_NAME
```

### 6.6.2 Reading from a File

To read data from a file, take the following steps:

1. Open the file—The OPEN/READ command opens the file for read access and associates the file name with a logical name.
2. Begin the read loop—File I/O is done in a loop unless you are reading or writing a single record.
3. Read the data from the file—Use the READ command with the /END\_OF\_FILE qualifier to read the next record in the file to a symbol. The /END\_OF\_FILE qualifier causes the VAX/VMS system to pass control to the label specified by the /END\_OF\_FILE qualifier if you have reached the end of the file. Generally, you specify the label that marks the end of the read loop.

If you are reading records from the terminal, the READ command automatically uses DATA: to prompt the user for a record. To specify a different prompt, use the /PROMPT qualifier of the READ command. The record value consists of all characters typed on the terminal in response to the prompt. Lowercase characters, multiple spaces, and tabs are preserved. Press CTRL/Z in response to the READ prompt to indicate the end of a file.

4. Process the data—Since you must read a file sequentially, process the current record before reading the next one.

5. Return to the beginning of the loop—You remain in the loop until you reach the end of the file.
6. End the loop and close the file—The CLOSE command disassociates the file name from the logical name and closes the file.

The following command procedure reads and processes each record from STAT.DAT.

```
$ OPEN/READ OUT_F STAT.DAT          !Open the file
$
$READ_DATA:                          !Begin the loop
$ READ/END_OF_FILE=END_READ OUT_F STUFF !Read a record and test for end of file
.                                     ! Process the data
.
$ GOTO READ_DATA                     !Go to the beginning of the loop
$END_READ:                          !End of loop
$
$ CLOSE OUT_F                        !Close the file
```

### 6.6.3 Modifying a File

You can modify a file in one of two ways:

- Rewrite records—This method allows you to make minor adjustments to one or more records in a file. You cannot change the size of a record or the number of records in the file.
- Rewrite the file—This method allows you to change, delete, and insert records. You create a new file using the old file as the main source of input.

#### 6.6.3.1 Minor Modifications

To make minor changes to the records in a file, take the following steps:

1. Open the file for both read and write access.
2. Use the READ command to read through the file until you reach the record that you want to modify.
3. Create a symbol containing the modified record. The modified record must be exactly the same size as the original record. If the text of the modified record is shorter, pad the record with spaces. If the text of the modified record is longer, you cannot use this method to modify the file.
4. Write the record.
5. Repeat steps 2 through 4 until you have changed all the records to be changed.

Since this method does not allow you to modify the size of the record, use it only if you have formatted the records in a file (for example, in a data file).



## 6-14 Command Procedures

The following command procedure reads each record in a data file. The record is displayed on the terminal, and the user is asked whether the record is to be modified. If the user chooses to modify the record, a new record is read from the terminal and its length is compared to the length of the original record. If the original record is longer, the new record is padded with spaces. If the original record is shorter, an error message is displayed, and the user is again prompted for a new record. If the user chooses not to modify the record, the next record is read from the file.

```
$ ! MODIFY.COM
$ !
$ ! Initialize string of spaces for padding
$ SPACES = "      "
$
$ ! Open the file
$ OPEN/READ/WRITE FILE STATS.DAT
$
$ ! Begin the loop
$BEGIN_LOOP:
$
$ ! Read and display a record
$ READ/END_OF_FILE=END_LOOP FILE RECORD
$PROMPT:
$ WRITE SYS$OUTPUT RECORD
$
$ ! Does the user want to change the record?
$ INQUIRE/NOPUNCTUATION YN "Change? [Y] "
$ ! if not, get next record
$ IF YN .EQS. "N" THEN GOTO BEGIN_LOOP
$
$ ! Otherwise, get the new record
$ INQUIRE NEW_RECORD "New record"
$
$ ! Compare the old and new records
$ OLD_LEN = F$LENGTH (RECORD)
$ IF OLD_LEN .GE. F$LENGTH(NEW_RECORD) THEN GOTO NO_ERROR
$ ! New record longer than old record
$ WRITE SYS$OUTPUT "ERROR -- New record is too long"
$ GOTO PROMPT
$NO_ERROR:
$ IF OLD_LEN .EQ. F$LENGTH(NEW_RECORD) THEN GOTO WRITE_RECORD
$ ! New record shorter than old record
$ PAD = F$EXTRACT(0,OLD_LEN-F$LENGTH(NEW_RECORD),SPACES)
$ NEW_RECORD = NEW_RECORD + PAD
$
```

```

$ ! Write the new record
$WRITE_RECORD:
$ WRITE/UPDATE FILE NEW_RECORD
$ GOTO BEGIN_LOOP
$
$END_LOOP:
$ CLOSE FILE

```

### 6.6.3.2 Major Modifications

To make extensive changes to a file, open that file for read access and open a new file for write access. Since the /WRITE qualifier opens a new file for write access, the new file can have the same name as the original file—the version number is incremented automatically.

**NOTE:** You must open the existing file for read access before you open the new version for write access to ensure that the correct file is opened for reading.

As you read each record from the original file, decide how the record is to be treated. In the following examples, the symbol RECORD contains the record read from the original file.

- No change—Write the same symbol to the new file.
 

```

$ ! No change
$ WRITE NEW_FILE RECORD

```
- Change—Use the INQUIRE command to read a different record into the symbol, then write the modified symbol to the new file.
 

```

$ ! Change
$ INQUIRE NEW_RECORD "New record"
$ WRITE NEW_FILE NEW_RECORD

```
- Delete—Do not write the symbol to the new file.
- Insert—Write the symbol to the new file. Then enter a loop that allows you to read a different record into the symbol and write the symbol to the new file.
 

```

$ ! Insertion
$ WRITE NEW_FILE RECORD
$
$LOOP:
$ !Get new records to insert
$ INQUIRE NEW_RECORD "New record"
$ IF RECORD .EQS. "" THEN GOTO END_LOOP
$ WRITE NEW_FILE NEW_RECORD
$ GOTO LOOP
$END_LOOP:

```

### 6.6.4 Handling I/O Errors

Use the `/ERROR` qualifier with the `OPEN`, `READ`, or `WRITE` command to suppress any error message and to pass control to a specified label. This qualifier overrides all other error-control mechanisms (except the `/END_OF_FILE` qualifier on the `READ` command). In the following command procedure, if an error occurs during execution of the `OPEN` command, the message "Error at OPEN" is printed and the procedure exits.

```
$ OPEN/READ/ERROR=READ_ERR OUT_F STAT.DAT
.
.
.
$ EXIT
$READ_ERR:
$ WRITE SYS$OUTPUT "Error at OPEN"
$ EXIT
```

## 6.7 Using Logic to Control the Flow of Execution

To create a command procedure that performs a consecutive series of DCL commands, type the commands in their order of execution. If you include a number of commands, group the commands and include comments explaining each group. To create a command procedure that requires decision making and/or variable input, the following guidelines are suggested.

### 6.7.1 Designing Complicated Command Procedures

Before writing a complicated command procedure, perform the tasks interactively. As you type the necessary commands, note the following:

- Variables—Data that changes each time you perform the task.
- Conditionals—Any command or set of commands that may vary each time you perform the task. Note the commands and the conditions under which you would execute it.
- Iteration—Any command or set of commands that you repeat. Note the commands and the factor that controls how often you repeat them.

The following example shows the commands needed to clean up a directory.



ZK-1750-84

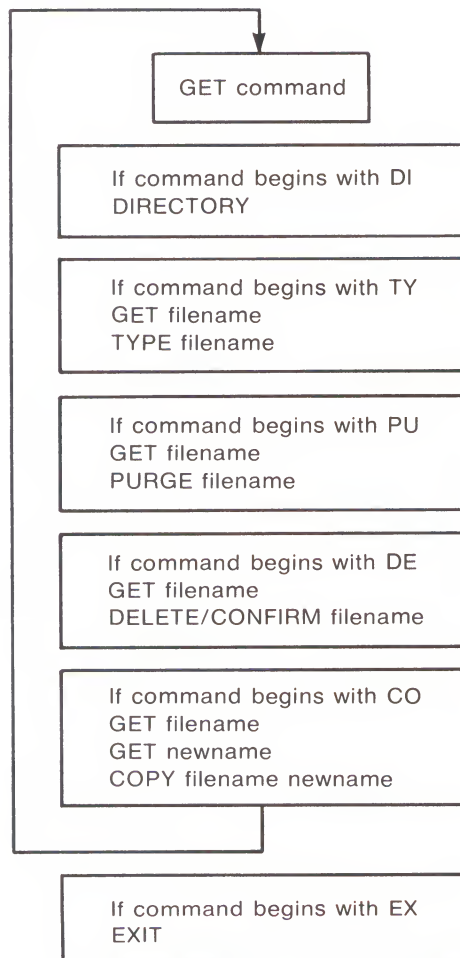
The file names change each time you clean your directory; therefore, they are variables. Any or all of the commands may be executed depending on the operation you need to perform; therefore, each command is conditional. The entire process is repeated until the directory is clean; therefore, it is iterative.

Decide how to load the variables, test the conditionals, and exit from the loop. The following decisions would be made for the clean directory procedure:

- Load variables—Get the file names from the terminal.
- Test conditionals—Get a command name from the terminal and execute the appropriate statements based on the command. The first two characters of each command must be read to differentiate between DELETE and DIRECTORY.
- Exit from loop—Get an EXIT command from the person using the procedure.

## 6-18 Command Procedures

Fill in the design:



ZK-1751-84

### 6.7.2 Coding Complicated Command Procedures

To make the command procedure easier to understand and to maintain, try to write the statements so that the procedure executes straight through from the first command to the last command. The following subsections describe how to execute



conditional code and loops in a linear fashion. Section 5.2 discusses the logical operators used in condition expressions.

### 6.7.2.1 Conditional Code

The recommended arrangement of conditional statements depends on whether a specified condition causes the execution of one or more commands.

- One command—To execute a single command, make the command part of the IF statement as shown. The command is executed if the condition is met.

IF condition THEN command

- More than one command—How you execute a block of commands depends on whether you leave the commands in the same command procedure or put them in another command procedure.

If you leave the commands in the command procedure, place them after the IF statement. If the condition is not met, pass control to a label at the end of the block of commands, as follows:

```
$ IF not condition THEN GOTO end_label
```

```
      .
      . ! block of commands
      .
```

```
$end_label:
```

If you put the commands into a separate command procedure, make the call to that command procedure part of the IF statement. The specified command procedure is executed if the condition is met.

```
IF condition THEN @command_procedure
```

In the following example, two commands (INQUIRE and PURGE) are executed if COMMAND equals "PU".

```
$ ! Purge a file
$ IF COMMAND .NES. "PU" THEN GOTO END_PURGE
$ INQUIRE FILESPEC "File to purge"
$ PURGE 'FILESPEC'
$END_PURGE:
```

```
      .
      .
      .
```

The following statement performs the same function. (The command procedure PURGE\_FILE.COM would be in a separate file; it is indented here for clarity.)

```
$ IF COMMAND .EQS. "PU" THEN @PURGE_FILE
    $ ! PURGE_FILE.COM
    $ ! Purge a file
    $ INQUIRE FILESPEC "File to purge"
    $ PURGE 'FILESPEC'
```

### 6.7.2.2 Case Statements

A case statement is a special form of conditional code that executes one out of a set of command blocks, depending on the value of a variable or expression. Typically, the valid values for the case statement are labels at the beginning of each command block. The case statement passes control to the appropriate block of code by using the specified value as the target label in a GOTO statement.

To write a case statement:

1. List the labels—Equate a symbol to a string that contains a list of the labels delimited by slashes (or any character you choose to act as a delimiter). This symbol definition should precede the command blocks.  

```
$ COMMAND_LIST = "/PURGE/DELETE/EXIT/"
```
2. Write the "case statement"—First get the value of the case variable. (Generally, the user supplies a value.) Then, use F\$LOCATE and F\$LENGTH (described in Appendix LEX) to determine whether the value of the case variable is valid. If so, pass control to the appropriate block of code. Otherwise, display a message and exit or request a different case value. (Since the label is equated to the full command name, F\$LOCATE includes the delimiters in its search for the command name to ensure that the command is not abbreviated.) The case statement should follow the list of labels and precede the blocks of code.

```
$GET_COMMAND:
$ INQUIRE COMMAND -
  "Command (EXIT,PURGE,DELETE)"
$ IF F$LOCATE ("/"+COMMAND+"/",COMMAND_LIST) .EQ. -
  F$LEN (COMMAND_LIST) THEN GOTO ERROR_1
$ GOTO 'COMMAND'

.
.
.

$ERROR_1:
$ WRITE SYS$OUTPUT "No such command as ''COMMAND'"
$ GOTO GET_COMMAND
```

3. Write the command blocks—Each block of commands may contain one or more commands. Begin each command block with a unique label. End each command block by passing control to a label outside the list of command blocks.

```

$GET_COMMAND:
    .
    .
    .
$PURGE:
$ INQUIRE FILE
$ PURGE 'FILE'
$ GOTO GET_COMMAND
$
$DELETE:
$ INQUIRE FILE
$ DELETE 'FILE'
$ GOTO GET_COMMAND
$
$EXIT:

```

### 6.7.2.3 Loops

The following arrangement is recommended for statements that form a loop:

1. Begin the loop.
2. Change the termination variable.
3. Test the termination variable. If the condition is met, go to the end of the loop.
4. Perform the commands in the body of the loop.
5. Return to the beginning of the loop.
6. End the loop.

You can also write loops that test the termination variable at the end of the loop rather than at the beginning, as follows:

1. Begin the loop.
2. Perform the commands in the body of the loop.
3. Change the termination variable.
4. Test the termination variable. If the condition is not met, go to the beginning of the loop.
5. End the loop.

Note that when you test the termination variable at the end of the loop the commands in the body of the loop are executed at least once regardless of the value in the termination variable.

## 6-22 Command Procedures

Both the examples that follow execute a loop that terminates when COMMAND equals "EX" (EXIT). (F\$EXTRACT, which is described in Appendix LEX, shortens COMMAND to its first two characters.) In the first example, COMMAND (the termination variable) is tested at the beginning of the loop; in the second, it is tested at the end.

```
$ ! EXAMPLE 1
$ !
$GET_COMMAND:
$ INQUIRE COMMAND-
  "Command (EXIT,DIRECTORY,TYPE,PURGE,DELETE,COPY) "
$ COMMAND = F$EXTRACT(0,2,COMMAND)
$ IF COMMAND .EQS. "EX" THEN GOTO END_LOOP
.
.   ! body of the loop
.
$ GOTO GET_COMMAND
$END_LOOP:

$ ! EXAMPLE 2
$ !
$GET_COMMAND:
$ INQUIRE COMMAND-
  "Command (EXIT,DIRECTORY,TYPE,PURGE,DELETE,COPY) "
$ COMMAND = F$EXTRACT(0,2,COMMAND)
.
.   ! body of the loop
.
$ IF COMMAND .NES. "EX" THEN GOTO GET_COMMAND
$ ! End of loop
```

To perform a loop a known number of times, use a counter as the termination variable. In the following example, 10 file names are input by the user and placed into the local symbols FIL1, FIL2, ..., FIL10:

```
$ NUM = 1                                ! Set counter
$LOOP:                                   ! Begin loop
$ INQUIRE FIL'NUM' "File"                ! Get file name
$ NUM = NUM + 1                           ! Update counter
$ IF NUM .LT. 11 THEN GOTO LOOP           ! Test for termination
$END_LOOP:                               ! End loop
.
.
.
```

To perform a loop for a known sequence of values, use F\$ELEMENT. In the following example, the files CHAP1, CHAP2, CHAP3, CHAPA, CHAPB, and CHAPC are processed in that order.

```
$ FILE_LIST = "1,2,3,A,B,C"
$ INDEX = 0
$PROCESS:
$ FILE = CHAP'F$ELEMENT(INDEX,"",FILE_LIST)'
$ IF FILE .EQS. "" GOTO END_LOOP

      . ! process file named by FILE
      .
$ INDEX = INDEX + 1
$ GOTO PROCESS
$ENDLOOP:
```

#### 6.7.2.4 Subroutines

The GOSUB command transfers control to a labeled subroutine in a command procedure without creating a new procedure level. Since the GOSUB command does not cause the creation of a new procedure level, it is referred to as a "local" subroutine call. The RETURN command terminates the GOSUB subroutine procedure, returning control to the command following the calling GOSUB statement.

The following command procedure shows how to use the GOSUB command to transfer control to labeled subroutines. After TEST2 is executed, the RETURN command returns control back to the command line following each calling GOSUB statement.

```
$!
$! GOSUB.COM
$!
$ SHOW TIME
$ GOSUB TEST1
$ WRITE SYS$OUTPUT "success completion"
$ EXIT
$!
$! TEST1 GOSUB definition
$!
$ TEST1:
$   WRITE SYS$OUTPUT "This is GOSUB level 1."
$   GOSUB TEST2
$   RETURN
$!
$! TEST2 GOSUB definition
$!
$ TEST2:
$   WRITE SYS$OUTPUT "This is GOSUB level 2."
$   RETURN
```



## 6-24 Command Procedures

The CALL command transfers control to a labeled subroutine in a command procedure and creates a new procedure level. Execution of the CALL command proceeds until an EXIT command is encountered. Control is then transferred to the command following the CALL command.

Local symbols and labels defined within a nested subroutine structure invoked with the CALL command are treated as if the routines had been invoked with the @ command. Labels are only valid for the subroutine level in which they are defined. Local symbols defined in an outer subroutine level are available to any inner subroutine levels.

The SUBROUTINE and ENDSUBROUTINE commands define the beginning and end of a subroutine invoked with the CALL command. The subroutine must begin with the SUBROUTINE command as the first executable statement. The ENDSUBROUTINE command functions as an EXIT command if an EXIT command is not specified in the procedure.

The label defining the entry point to the subroutine must appear either before the SUBROUTINE command or on the same command line. A subroutine can have only one entry point.

The following procedure shows how to use CALL to transfer control to a labeled subroutine. The example also shows that you can call another command file from within a subroutine. (You can also call another subroutine from a subroutine.) The CALL command invokes the subroutine SUB1, directing output to the file NAMES.LOG and allowing other users write access to the file.

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

```

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

```

### 6.7.3 Testing and Debugging

For lengthy or complicated command procedures, write the logic for the main procedure, but use stubs for the nested procedures and subroutine-type pieces of code. A stub writes a message stating the function it is replacing.

```

.
.
.
$ ! Purge a file
$ IF COMMAND .NES. "PU" THEN GOTO END_PURGE
$ WRITE SYS$OUTPUT "Purge routine" ! stub
$END_PURGE:
.
.
.

```

If you have a number of places that need stubs, you can use one nested command procedure to insert the stub logic. (The command procedure STUB.COM would be in a separate file; it is indented here for clarity.)

```

.
.
.
$ ! Purge a file
$ IF COMMAND .NES. "PU" THEN GOTO END_PURGE
$ @STUB "Purge"
    $ ! STUB.COM
    $ ! Procedure STUB
    $ WRITE SYS$OUTPUT "'P1' routine"
$END_PURGE:
.
.
.

```

## 6-26 Command Procedures

Once you have written the code using stubs, you can test the overall logic of the command procedure. Test all possible paths of execution. For example, to test the following statements, print the purge and the delete messages and then exit using the EXIT command.

```
$ ! CLEAN.COM
$ !
$GET_COMMAND:
$ ! Read a command from the terminal
$ INQUIRE COMMAND-
  "Command (EXIT, DIRECTORY, TYPE, PURGE, DELETE, COPY)"
$ COMMAND = F$EXTRACT(0,2,COMMAND)
$
$ IF COMMAND .EQS. "EX" THEN GOTO END_COMMAND
$
$ ! Purge a file
$ IF COMMAND .NES. "PU" THEN GOTO END_PURGE
$ WRITE SYS$OUTPUT "Purge routine."
$END_PURGE:
$
$ ! Delete a file
$ IF COMMAND .NES. "DE" THEN GOTO END_DELETE
$ WRITE SYS$OUTPUT "Delete routine."
$END_DELETE:
.
.
.
$ GOTO GET_COMMAND
$END_COMMAND:
```

Once the overall logic of the procedure works, you can begin filling in the stubs. Fill in the first stub, test it, and debug it if necessary. When that stub works, move on to the next one.

The following commands are useful for debugging command procedures:

- SET VERIFY—SET VERIFY prints each line before it is executed. When an error occurs with verification set, you see the error and the line that generated the error. In the following example, seeing the command line that generated the error explains the error message.

```

$ SET VERIFY
$ @CDIR
$ ! Read a command from the terminal
$ INQUIRE COMMAND-
  "Command (EXIT, DIRECTORY, TYPE, PURGE, DELETE, COPY)"
Command (EXIT, DIRECTORY, TYPE, PURGE, DELETE, COPY): DELETE
$ COMMAND = F$EXTRACT(0,2,COMMAND)
$GET_COMMAND:
$ IF COMMAND .EQ. "EX" THEN GOTO END_COMMAND
%DCL-W-IVCHAR, non-numeric character in value string
\E\ EXIT
$ IF COMMAND .NES. "DI" THEN GOTO END_DIR

```

The logical operator .EQ. is used to compare numbers, not strings (see Section 5.2). To correct the error, change .EQ. to .EQS.. Note that you can use keywords with SET VERIFY to indicate that only command lines or data lines are to be verified.

- **SHOW SYMBOL**—Use the SHOW SYMBOL command to print the values of the symbols involved in an error. In the following procedure, the IF statements are not passing control to the expected procedures. Putting the SHOW SYMBOL COMMAND command before the IF statements allows you to check the value of COMMAND.

```

$ SET VERIFY
$ @CDIR
$GET_COMMAND:
$ ! Read a command from the terminal
$ INQUIRE COMMAND-
  "Command (EXIT, DIRECTORY, TYPE, PURGE, DELETE, COPY)"
Command (EXIT, DIRECTORY, TYPE, PURGE, DELETE, COPY): DELETE
$ COMMAND = F$EXTRACT(1,2,COMMAND)
$ SHOW SYMBOL COMMAND
  COMMAND = "EL"
$GET_COMMAND:
$ IF COMMAND .EQS. "EX" THEN GOTO END_COMMAND

```

The F\$EXTRACT lexical function is extracting two characters beginning at character 1 (the second character) rather than at character 0 (the first character). To correct the error, change F\$EXTRACT(1,2,COMMAND) to F\$EXTRACT(0,2,COMMAND). Note that INQUIRE automatically converts input to uppercase; therefore, the quoted string

in the IF statement must be written in uppercase for DCL to evaluate the strings as equal.

## 6.8 Handling Errors and CTRL/Y Interrupts

The following table describes the default action taken when an error occurs or when you press CTRL/Y. These default actions can be overridden with the ON, SET [NO]ON, and SET [NO]CONTROL=Y commands.

Interrupt	Default Action
Error or severe error	Procedure exits to the next command level.
CTRL/Y at DCL command level or command level 1	Interrupts procedure: procedure can continue if no other image forces it to exit.
CTRL/Y at command level lower than level 1	Procedure exits to the next higher command level.

### 6.8.1 The ON Command

The ON command specifies an action to be performed if an error of a particular severity or greater occurs. See Section 1.9 for a discussion of error conditions and severity levels. If such an error occurs, the system takes the following actions:

1. The error message is displayed.
2. The action specified by the ON command is performed.
3. The default error action (exit to the next higher command level) is reset.

If an error of less than the specified severity occurs, the error message is displayed and the command procedure continues executing. Assume a command procedure executes the following command:

```
$ ON ERROR THEN GOTO ERR1
```

The command procedure continues to execute unless an error or severe error occurs. If such an error occurs, the error message is displayed; the default error action (exit to the next higher command level) is reset; and the command procedure continues execution at ERR1. If a second error occurs before another ON or SET NOON command is executed, the procedure exits to the next higher level.



The execution of an ON statement performs an implicit SET ON function, thus nullifying any SET NOON condition that may be in effect.

**NOTE:** Only one ON statement can be in effect at any one time. If more than one ON statement has been issued, only the last one is in effect.

### 6.8.2 The SET [NO]ON Command

The SET ON and SET NOON commands enable and disable error checking for the current command level. The SET NOON command overrides the ON command. If an error (regardless of severity) occurs after a SET NOON command is executed, the system takes the following actions:

1. Displays the error message
2. Continues execution of the procedure

SET NOON remains in effect until either an ON or SET ON command is executed. The SET ON command resets the action specified by the last ON command.

In the following command procedure, if an error or severe error occurs while copying the file, the procedure continues to execute without going to GET\_COMMAND as specified by the ON command.

```
$ ON ERROR THEN GOTO GET_COMMAND
$GET_COMMAND:
.
.
.
$ ! Type a file
$ IF COMMAND .NES. "CO" THEN GOTO END_COPY
$ INQUIRE FILESPEC "File to move"
$ INQUIRE COPYSPEC "New file specification"
$ SET NOON
$ COPY 'FILESPEC' 'COPYSPEC'
$ SET ON
$END_COPY:
.
.
.
```

### 6.8.3 CTRL/Y Interrupts

The ON CONTROL\_Y command specifies an action to be performed when CTRL/Y is pressed at the command level that defined the CTRL/Y action. In the following command procedure, pressing CTRL/Y while a file is being typed passes control to END\_TYPE.

```

.
.
.
$ ! Type a file
$ IF COMMAND .NES. "TY" THEN GOTO END_TYPE
$ ON CONTROL_Y THEN GOTO END_TYPE
$ TYPE 'FILESPEC'
$END_TYPE:
$
$ !Reset default
$ SET NOCONTROL=Y
$ SET CONTROL=Y
.
.
.

```

An ON CONTROL\_Y command remains in effect until another ON CONTROL\_Y or a SET NOCONTROL=Y command is executed or the command procedure exits.

See Section 6.9 for another example of using the ON CONTROL\_Y command.

To exit from a nonterminating loop when CTRL/Y is disabled, you must delete your process from another terminal using the DCL command STOP. If you disable the CTRL/Y default action, reset it as soon as possible. To reset the CTRL/Y default, execute the SET NOCONTROL=Y command followed by the SET CONTROL=Y command, as shown in the previous example.

## 6.9 Cleanup Operations

In general, execution of a command procedure should not affect the user's process state. Therefore, a command procedure should include a set of commands that returns the process to its original state. Common cleanup operations include the following (see Appendix LEX for lexical function specifications):

- Closing files—If you have opened any files, make sure that they are closed before the command procedure exits. You can use the lexical function F\$GETJPI to examine the remaining open file quota (FILCNT) for the process. If FILCNT is the same at the beginning and end of the command procedure, you know that no files have been left open. In the following example, a warning message is displayed if a file is left open.

```
$ FIL_COUNT = F$GETJPI("", "FILCNT")
```

```
$ IF FIL_COUNT .NE. F$GETJPI("", "FILCNT") THEN-
  WRITE SYS$OUTPUT "WARNING -- file left open"
```

- Deleting temporary or extraneous files—If you have created temporary files, delete them. In general, if you have updated any files, you should purge them to delete the previous copies. Take care in deleting files that you have not created. For example, if you have updated a file that contains crucial data, you may wish to make the purging operation optional.
- Resetting default device and directory—If you change the default device and/or directory, reset the original defaults before the command procedure exits.

To save the name of the original default directory, use the DEFAULT keyword of the F\$ENVIRONMENT lexical function. At the end of the command procedure, include a SET DEFAULT command that restores the saved device and directory. The following example saves and restores device and directory defaults.

```
$ SAV_DEFAULT = F$ENVIRONMENT("DEFAULT")
```

```
$ SET DEFAULT 'SAV_DEFAULT'
```

The following table lists other commonly changed process characteristics along with the lexical functions and commands used to save and restore the original settings.

Characteristic	To save...	To restore...
DCL prompt	F\$ENVIRONMENT	SET PROMPT
Default protection	F\$ENVIRONMENT	SET PROTECTION/DEFAULT
Privileges	F\$PRIVILEGES	F\$SETPRV or SET PROCESS /PRIVILEGES
Control characters	F\$ENVIRONMENT	SET CONTROL
Verification	F\$ENVIRONMENT	F\$VERIFY
Message format	F\$ENVIRONMENT	SET MESSAGE
Key state	F\$ENVIRONMENT	SET KEY

To ensure that cleanup operations are performed even if the command procedure is aborted, begin the command procedure with the following statement:

```
$ ON CONTROL_Y THEN GOTO CLEAN_UP
```

Cleanup operations follow the CLEAN\_UP label. Default CTRL/Y action is reset when the command procedure exits.



## Chapter 7

# Accounts and Security

### 7.1 User Accounts

User accounts are maintained in a file named SYS\$SYSTEM:SYSUAF.DAT, referred to as the user authorization file, or UAF. A VAX/VMS system uses the UAF to validate login requests and to set up processes for users who successfully log in. You can examine and modify this file with the Authorize Utility (AUTHORIZE).

#### 7.1.1 User Authorization File

The UAF contains a record for each account. Each record consists of fields providing information for the following areas: identification, login characteristics, login restrictions, priority, limits, and privileges. The username field is specified as a parameter to Authorize Utility commands; the other fields are specified as qualifier values of Authorize Utility commands. See the *VAX/VMS Authorize Utility Reference Manual* for descriptions of the qualifiers and information on invoking the Authorize Utility.

### 7.2 Protection

A VAX/VMS system provides two related mechanisms to control the access that users have to system objects.

- UIC-based protection—Each user process in the system is assigned a user identification code (UIC) in the user authorization file (UAF) with the Authorize Utility. Each object on the system is also associated with a UIC (typically the UIC of its creator). A protection mask associated with an object determines the type of access allowed to a user, based on the relationship between the user UIC and the object UIC.



## 7-2 Accounts and Security

- **ACL-based protection**—An access control list (ACL) that specifies the type of access to be granted or denied to a particular user or group of users may be associated with a system object. The system objects for which ACL-based protection may be specified are: files, directories, devices, logical name tables, and global sections. Users are specified by identifiers in the rights database that are assigned with the Authorize Utility.

The system determines whether to grant a user access to an object in the following steps:

1. **ACL**—If the user matches an identifier in the object's ACL, the system grants or denies access based on the ACL. However, even if an entry in the ACL denies access, the system may still grant access based on the SYSTEM and OWNER fields of the UIC-based protection.
2. **UIC**—If the user does not match an identifier in the object's ACL or the object has no ACL, the system grants or denies access based on the relationship between the user's UIC and the object's UIC as qualified by the object's protection mask.
3. **Privileges**—If the system denies the user access, the user may be granted access by using one of the following privileges: BYPASS, GRPPRV, READALL, or SYSPRV.

UIC-based protection is useful for denying or granting access to a specified group of users or to all users on the system. The optional ACL-based protection allows further control over the protection of an object: you can grant or deny access to individual users, and you can further identify users by certain aspects of their usage (such as whether they are interactive, batch, local, remote, or dial-up users). The combination of UIC and ACL protection provides a way to specify multiple subsets and overlapping groups of users.

### 7.2.1 UIC-Based Protection

Typically, you use UIC-based protection if the object is to be accessed by (1) only the owner, (2) all users on the system, or (3) a specific group of users.

### 7.2.1.1 User Identification Code

UICs are a subset of the identifiers that the VAX/VMS system uses to identify users and groups of users (see Section 7.2.2.2 for other identifiers). A UIC consists of two parts, group and member, specified in the following format:

[group,member]

A UIC can be either numeric or alphanumeric.

- **Numeric UIC**—Consists of a group number in the range 0 through 37776 (octal) and a member number in the range 0 through 177776 (octal).
- **Alphanumeric UIC**—Consists of a member name (the username parameter specified with the Authorize Utility command ADD) and, optionally, a group name (the name specified with the Authorize Utility command ADD /ACCOUNT); both member and group names must contain at least one alphabetic character and up to a maximum of 31 alphanumeric characters (including A–Z, 0–9, underscore, and dollar sign characters). An alphanumeric UIC is equated to a numeric UIC in the rights database by default once the rights database has been created. You can generally specify a numeric UIC and its equivalent alphanumeric UIC interchangeably.

The member component of a UIC must be unique to the system. By default, the member component of an alphanumeric UIC is equated to both the group and member components of a numeric UIC in the rights database (so that specifying just the member part of an alphanumeric UIC is sufficient). The following examples illustrate several UICs in proper UIC notation.

UIC	Meaning
[200,10]	Group 200, member 10
[3777,3777]	Group 3777, member 3777
[USER,FRED]	Group USER, member FRED
[EXEC,JONES]	Group EXEC, member JONES
[JONES]	Group EXEC, member JONES

When you log in to a VAX/VMS system, your process UIC is specified in your UAF account. Typically, your process UIC does not change, although it can be changed with the SET UIC command (which requires CMKRNL privilege). By default, detached processes (created by the DCL command SUBMIT or RUN) and subprocesses (created by the DCL command SPAWN) take the same UICs as their creators. If you have DETACH privilege, you can create a detached process with a different UIC (by using the /UIC qualifier of the RUN command).

By default, an object (such as a file) receives the UIC of the process creating it. You can specify the UIC of a file with the `/OWNER_UIC` qualifier of the `BACKUP`, `CREATE`, `SET DIRECTORY`, and `SET FILE` commands. With `SYSPRV` privilege, you can specify any UIC; with `GRPPRV` privilege, you can specify any member within your current group; otherwise, you can specify only your own UIC. You can specify the UIC of a disk volume with the `/OWNER_UIC` qualifier of the `INITIALIZE` and `MOUNT` commands (except for the system disk, which retains the UIC specified when it was initialized). You can also change the UIC of a disk (including the system disk) with the `SET VOLUME` command. You must have the `VOLPRO` privilege to specify a UIC other than your own. In addition, when you initialize a system disk (`/SYSTEM` qualifier), it receives a UIC of `[1,1]` and a group disk (`/GROUP` qualifier) receives a UIC of `[n,0]`, where `n` is the group number of the owner. You can specify a UIC for a device such as a terminal (by default, devices are not owned) with the `/OWNER_UIC` qualifier of the `SET PROTECTION/DEVICE` command.

### 7.2.1.2 Ownership and Access Categories

The relationships between the UIC of a process and the UIC of an object fall into the following four ownership categories:

- **System**—The UIC of the process is in the range 1 through 10 (octal) or the process has `SYSPRV` (or `GRPPRV`) privilege. (The range of system UICs is determined by the `SYSGEN` parameter `MAXSYSGROUP`, which defaults to 10 octal.)
- **Owner**—The UIC of the process and the UIC of the object are identical.
- **Group**—The group number of the process and the group number of the object are identical.
- **World**—The UIC of the process and the UIC of the object can have anything or nothing in common.

A process and object may have any number of the above relationships. For example, a process that owns an object necessarily has group and world relationships with the object.

A process can access an object in the following ways:

- **Read (allocate)**—Read a file; read from a disk volume; allocate nonfile devices.
- **Write**—Write a file; write to a disk volume.
- **Execute (create)**—Execute an image file; look up entries in a directory if you explicitly specify the file name (without using wildcard characters); create files on a disk volume.
- **Delete**—Delete files.

### 7.2.1.3 Protection Masks

A protection mask consists of four fields, each with four indicators. Each field applies to one category of ownership. Each indicator within a field applies to one category of access. The fields and indicators are as follows:

Ownership	Access			
SYSTEM	READ	WRITE	EXECUTE	DELETE
OWNER	READ	WRITE	EXECUTE	DELETE
GROUP	READ	WRITE	EXECUTE	DELETE
WORLD	READ	WRITE	EXECUTE	DELETE

Protection for an object must be specified in the following format:

(ownership[:access],...)

Specify ownership as one of the following (each may be abbreviated to one character): SYSTEM, OWNER, GROUP, or WORLD. Specify access as one or more of the following indicators: R (read), W (write), E (execute), D (delete). Omission of the colon and access indicators means no access for that category of ownership. The following protection specification allows system users full access to an object, the owner full access except delete, and group and world users no access:

(S:RWED,O:RWE,G,W)

Omission of an ownership category generally results in no access being granted that category. However, the SET PROTECTION command retains the existing protection of the object for omitted fields of the ownership category.

### 7.2.1.4 Securing User Data and Devices

The suggestions for UIC-based protection of data and devices belonging to individual and application accounts are as follows:

- **Default protection**—Make sure your default protection is adequate for the average situation. In general, you do not want to grant write or delete access to world users. You may or may not want to grant write access to group users. You may or may not want to grant read access to world users.
- **Sensitive files**—Protect sensitive files by specifying extra protection, for example, with the SET PROTECTION command. You can also protect sensitive files by maintaining them in a subdirectory on which extra protection is set. (However, to protect sensitive files completely, the directory protection alone is not adequate. You must also protect each individual file contained within the directory. You can add a default protection ACE to the subdirectory file to facilitate this process.



## 7-6 Accounts and Security

- Individual access—Use access control lists (ACLs) to grant or deny individual users access to a file (see Section 7.2.2).
- Copied files—In general, do not copy a file into someone else's directory (for example, if you have write access to a group member's directory), as it will have your UIC instead of the UIC of the directory's owner. Use the MAIL command to send the file, or have the owner of the directory copy the file.
- Private volumes—If you mount a private volume, be sure to deallocate the device when you are done.

### 7.2.2 ACL-Based Protection

Typically, you use access control lists (ACLs) on system objects to grant or deny access to individual users, groups of users, and to subsets of user groups. ACLs contain entries (ACEs) that specify the access to be granted or denied a user or group of users. The user or group of users is designated by identifiers.

#### 7.2.2.1 Object Types

You can establish ACLs for various system objects: files, directory files, global sections, devices, and logical name tables. In general, you need not be concerned about the object type when establishing or changing an ACL; however, you do need to be aware of few object-specific considerations. This section describes these considerations.

##### Devices

You must reestablish ACLs on devices every time the system is booted because they are not saved. ACLs on devices are set up and modified with the ACL Editor or with the command:

```
$ SET ACL/OBJECT_TYPE=DEVICE device-name
```

##### Global Sections

You must reestablish ACLs on global sections (except those backed by files) every time the system is booted because they are not saved.

The ACL on a global section backed by a file is the ACL of the file. Changing the file's ACL causes the corresponding change in the global section's ACL. The ACL on the global section itself cannot be changed directly.

You can establish ACLs on both system and group global sections. Note, however, that if a user attempts to access a group global section that is outside his UIC group, the operating system denies him access and does not consider the ACL. ACLs on



page frame numbers (PFN) and page file global sections may be set up and modified with the ACL Editor or with the commands:

```
$ SET ACL/OBJECT_TYPE=SYSTEM_GLOBAL_SECTION section_name
$ SET ACL/OBJECT_TYPE=GROUP_GLOBAL_SECTION section_name
```

### Logical Name Tables

You must reestablish ACLs on logical name tables every time the system is booted because they are not saved. ACLs on logical name tables are set up and modified with the ACL Editor or with the command:

```
$ SET ACL/OBJECT_TYPE=LOGICAL_NAME_TABLE table-name
```

### 7.2.2.2 Identifiers

An identifier is a value that represents an individual user, a group of users, or an aspect of the user's environment. There are three types of identifiers:

- UIC
- General identifiers
- System-defined identifiers

UICs can be in both numeric and alphanumeric form and are (see Section 7.2.1.1) useful in identifying individual users. UIC identifiers must be enclosed in brackets and may have wildcard characters in either the group or member fields (for example, [EXEC,\*]).

Identifiers include other general identifiers that you explicitly associate with users in the rights database. These general identifiers are useful in identifying multiple groups of users outside the bounds of UIC groups. For example, you could create the identifier SECRET and assign it in the rights database to a selected group of users, some of whom could be in different UIC groups.

A third type of identifier includes the following system-defined identifiers, which you can use to identify users by their mode of using the system.

System-defined Identifiers	Type of user
BATCH	Batch user
DIALUP	User logged in on a dial-up terminal
INTERACTIVE	Interactive user
LOCAL	User logged in on local terminal
NETWORK	Network process
REMOTE	User logged in over the network

## 7-8 Accounts and Security

Generally, you should treat the preceding system-defined identifiers as being mutually exclusive. However, you can combine them with UIC identifiers or general identifiers by connecting them with plus signs (for example, [FRED]+BATCH). Access is granted only if both identifiers are true. (In the example [FRED]+BATCH, the user identified as [FRED] must be running a batch job for the system to grant the specified access.)

The following are examples of user-defined identifiers that are valid for ACL-based protection:

- PAYROLL—Specifies all users holding the identifier PAYROLL.
- [USER,JONES]—Specifies the user whose alphanumeric UIC is group USER and member JONES.
- [200,10]—Specifies the user whose numeric UIC is group 200, member 10.
- [FRED]+BATCH—Specifies the batch user whose alphanumeric UIC is FRED.
- DIALUP—Specifies all users logged in on a dial-up terminal.

### 7.2.2.3 Access Control Entries (ACE)

An entry in an access control list specifies the access to a system object that is to be granted or denied a user. This access is specified by the identifier. Different kinds of access are: NONE, READ, WRITE, EXECUTE, DELETE and/or CONTROL.

There are three types of ACEs:

- Identifier ACE
- Default protection ACE
- Security alarm ACE

### 7.2.2.4 IDENTIFIER ACEs

An identifier ACE controls the type of access allowed to a particular user or group of users. Identifier ACEs have the following format:

(IDENTIFIER=identifier[,OPTIONS=options+...],ACCESS=access+...)

The following are examples of ACEs:

(IDENTIFIER=[200,201],ACCESS=READ+WRITE+EXECUTE)

Grants the user identified by UIC identifier [200,201] read, write, and execute access to the system object.

(IDENTIFIER=[FRED]+BATCH,ACCESS=WRITE+EXECUTE)

Grants batch users with the alphanumeric UIC [FRED] write and execute access to the system object.

(IDENTIFIER=PAYROLL,ACCESS=READ)

Grants users who hold the identifier PAYROLL read access to the system object.  
(IDENTIFIER=DIALUP,ACCESS=NONE)

Denies holders of the system-defined identifier DIALUP any access to the system object.

The preceding ACEs could be specified in a single ACL for a system object, as follows:

```
(IDENTIFIER=[200,201],ACCESS=READ+WRITE+EXECUTE)
(IDENTIFIER=[FRED]+BATCH,ACCESS=WRITE+EXECUTE)
(IDENTIFIER=PAYROLL,ACCESS=READ)
(IDENTIFIER=DIALUP,ACCESS=NONE)
```

To specify one or more default ACEs for inclusion in the ACLs of files subsequently created in a directory, use the OPTIONS=DEFAULT option of an identifier ACE. The following ACE, when placed in the ACL of the directory file, grants all users holding the SECRET identifier read, write, and execute access to new files in the directory by default.

```
(IDENTIFIER=SECRET,OPTIONS=DEFAULT,ACCESS=READ+WRITE+EXECUTE)
```

**NOTE:** Default protection is associated only with newly created files, not existing ones.

### 7.2.2.5 Rights List

The system determines protection by checking the object's ACL against the list of identifiers held by the accessor to find a matching entry. This list, called a **rights list**, is maintained in the **rights database**.

The rights list is the portion of the rights database associated with a specific user. The rights list is created for each user at login and contains all the identifiers and attributes held by the user. On the other hand, the rights database is a file that contains ALL the identifiers defined in the system.

You can add or remove entries in the rights list with the DCL command SET RIGHTS\_LIST. This differs from the rights database, which requires you to use AUTHORIZE commands.

Beginning with the first entry in the rights list, all the accessor's identifiers are compared to each entry in the list; the accessor must hold all identifiers specified in a single entry for a match to exist. The search continues until a match is found.

Because the system determines access at the first matching entry, the order of the entries is critical. For example, if the last entry in the previous example—(IDENTIFIER=DIALUP,ACCESS=NONE)—were placed at the top of the list, all dial-up users (including those specified in the remaining entries) would be denied access to the associated file. Placing it last in the access control list allows users holding identifiers [200,201], [FRED]+BATCH, and [PAYROLL] the specified access, even when they are dial-up users.

### 7.2.2.6 DEFAULT\_PROTECTION ACES

To specify default protection for new files in a particular directory, place a default\_protection ACE in the ACL of the directory file. (The directory file is the file with the directory name as file name and DIR as file type in the parent directory). The default\_protection ACE affects files that are subsequently created in the directory and in any subdirectories under that directory unless protection is specified for one of those files individually. Default\_protection ACEs apply UIC-based protection. Specify a default\_protection ACE in the following format, where *protection-mask* is the same mask used in UIC protection (see Section 7.2.1.3):

(DEFAULT\_PROTECTION[,options],protection-mask)

The following default\_protection ACE specifies that by default the system and owner have read, write, execute, and delete access to any files subsequently created for the directory and that group and world users have no access.

(DEFAULT\_PROTECTION,S:RWED,O:RWED,G,W)

### 7.2.2.7 ALARM\_JOURNAL ACES

The alarm\_journal ACE allows you to specify that an alarm message be sent to the security operator's terminal if a certain type of access takes place. Alarms are supported only for files and global sections. The alarm\_journal ACE functions only when alarms to the security operator's terminal have been enabled through the DCL command SET AUDIT, security messages to the operator's terminal have been enabled with the DCL command REPLY/ENABLE=SECURITY, and the OPCOM process is executing.

The format of an alarm\_journal ACE is

(ALARM\_JOURNAL=SECURITY[,options+...][,access+...])

The following alarm\_journal ACE specifies that an alarm will be sent to the security operator's terminal if an accessor attempts to read the object, and that this ACE will be preserved even when an attempt is made to delete the entire ACL.

(ALARM\_JOURNAL=SECURITY,OPTIONS=PROTECTED,ACCESS=READ+SUCCESS+FAILURE)

For an alarm to have any effect, you must include either SUCCESS or FAILURE or both in the ACCESS field.

### 7.2.3 Protection of Files

For the most part, file protection is transparent. The tools exist, however, to adjust the protection of a file as you see fit. You must own the file, have control access to the file, or have GRPPRV, SYSPRV, BYPASS, or READALL privilege to set the protection or modify the ACL of a file.

**NOTE:** You cannot completely protect a file without applying at least the same protection to the directory in which the file resides; see Section 7.2.3.3 for information on directory protection.

#### 7.2.3.1 Default File Protection

A new file receives default UIC-based protection and the default ACEs (if any) of its parent directory. A renamed file's protection is unchanged. A new version of an existing file receives the UIC-based protection and ACL of the previous file. (Use the /PROTECTION qualifier of the BACKUP, COPY, and CREATE commands to override the default UIC-based protection.)

- **Default UIC protection**—The operating system provides each process with a default UIC-based protection of (S:RWED,O:RWED,G:RE,W). To change the default protection, invoke the SET PROTECTION command with the /DEFAULT qualifier. For example, if you place the following command in your login command procedure, you grant all processes read and execute access to any files that you subsequently create. (Remember that you must execute the login command procedure for this command to execute.)

```
$ SET PROTECTION = (S:RWED,O:RWED,G:RE,W:RE)/DEFAULT
```

- **Default ACL protection**—You can override default UIC protection for specified directories or subdirectories by placing a default\_protection ACE in the ACL of the appropriate directory file. The default protection specified in the ACE is applied to any new file created in the specified directory or any subdirectory of the directory. The following ACE, which must be in the ACL of a directory file, specifies that the default protection for that directory and the directory's subdirectories allow system and owner processes full access, group processes read and execute access, and world users no access.

```
(DEFAULT_PROTECTION,S:RWED,O:RWED,G:RE,W:)
```

To specify a default identifier ACE to be copied to the ACL of any file subsequently created in the directory, specify the DEFAULT option in the directory file's identifier ACL.



### 7.2.3.2 Explicit File Protection

You can explicitly specify UIC-based protection for a new file with the /PROTECTION qualifier (valid with the BACKUP, COPY, and CREATE command), as demonstrated:

```
$ CREATE MAST12.TXT/PROTECTION=(S:RWED,O:RWED,G,W)
```

You can change the UIC-based protection on an existing file with the SET PROTECTION command, as demonstrated:

```
$ SET PROTECTION=(S:RWED,O:RWED,G,W) MAST12.TXT
```

After a file is created and you have created an ACL for the file, you can modify the ACL and add as many ACEs to the ACL as you desire. The protection specified by the ACL overrides the file's UIC protection.

### 7.2.3.3 Protection of Directories

You cannot completely protect a file without applying at least the same protection to the directory in which the file resides. For example, if you deny a user all access to a file but allow that user read access to the file's directory, the user cannot access the contents of the file but can see that it exists. Conversely, a user allowed access to a file and denied access to the file's directory (or one of the parent directories) cannot see that the file exists.

**NOTE:** To protect sensitive files, the directory protection alone is not adequate. You must also protect each individual file contained within the directory.

By default, top-level directories receive UIC-based protection (S:RWE,O:RWE,G:RE,W:E) and no ACL. Subdirectories receive UIC-based protection minus any delete access or default\_protection ACEs from the parent directory.

To specify UIC-based protection explicitly when creating a directory, use the /PROTECTION qualifier of the CREATE/DIRECTORY command. You cannot specify an ACL for the directory until the directory is created. To change the UIC-based protection of an existing directory, use the SET PROTECTION command (apply this command to the directory file). To specify or change the ACL of an existing directory, edit the directory file's ACL (see Section 7.2.3.1).

You can limit but not prohibit directory access by specifying execute access but not read access. Execute access on a directory permits you to examine and read files that you know are contained in the directory (that is, you know the file specifications) but prevents you from displaying a list of the files in the directory.

### 7.2.3.4 Protection of Mail Files

Mail files receive the protection (S:RWD,O:RW,G,W). Files of type MAI created with the EXTRACT command of the Mail Utility receive the protection (S:RWD,O:RWD,G,W).

## 7.2.4 Protection of Disk Volumes

By default, no protection is applied to newly initialized disk volumes. You can specify protection with the `/PROTECTION` qualifier of the `INITIALIZE` command and you can specify an ACL for a disk volume. The following example specifies UIC-based protection for the disk volume `ACCOUNT1`.

```
$ INITIALIZE WORKDISK: ACCOUNT1 -
_$ /PROTECTION=(S:RWED,O:RWED,G:R,W:R)
```

You can respecify the protection each time you mount the volume with the `/PROTECTION` qualifier of the `MOUNT` command. You must own the volume or have `VOLPRO` privilege to change protection.

You can also limit access to a disk volume with the following qualifiers to the `INITIALIZE` and `MOUNT` commands:

- `/SYSTEM`—All processes have `RWED` access to the volume, but only system processes (or processes with `SYSNAM` and `SYSPRV` privileges) can create first-level directories. (The volume is owned by [1,1].)
- `/GROUP` and `/NOSHARE`—System, owner, and group processes have `RWED` access to the volume. World users have no access.
- `/NOSHARE`—System and owner processes have `RWED` access to the volume. Group and world users have no access.

At initialization time, the above qualifiers override any protection mask specified. At mount time, however, the protection mask overrides the qualifiers. When mounting a volume, you must have `GRPNAM` privilege to specify `/GROUP` and `SYSNAM` privilege to specify `/SYSTEM`.

## 7.2.5 Protection of Devices

In general, device protection controls the ability to allocate the device and is specified by granting read access in an ACL. To specify an ACL for a disk device, use the `SET ACL/OBJECT_TYPE=DEVICE` command. For example, to grant a user with the alphanumeric UIC `[FRED]` read access to the disk device `WORKDISK`, type:

```
$ SET ACL/OBJECT_TYPE=DEVICE/ACL=(IDENTIFIER=[FRED],ACCESS=READ) -
_$ WORKDISK
```

Note that when you specify an ACL for a disk device, the ACL is associated with the device, not with the disk volume. For example, if you mount a disk device on `WORK1`, specify the preceding `SET ACL/OBJECT_TYPE=DEVICE` command, and then dismount the disk device, the ACL protection remains on `WORK1` but not on the disk device.

The only protection that applies to a nonfile device is the ability to allocate it, specified by read access. By default, nonfile devices such as mailboxes are unprotected. Interactive terminals are set up to provide complete access to system users and no access to all other users. (Note that access here refers to access via an application program. The device protection on a terminal does not control who can log in on it.) You can change the protection of a nonfile device through the use of ACLs or by changing the standard UIC protection. Modify the ACL with the DCL command SET ACL/OBJECT\_TYPE=DEVICE. Modify the UIC protection with the DCL command SET PROTECTION/DEVICE (requires OPER privilege). For example, the following command allows users holding the PAYROLL identifier read access to TERMINAL3.

```
$ SET ACL/OBJECT_TYPE=DEVICE/ACL=(IDENTIFIER=PAYROLL,ACCESS=READ)
_$ TERMINAL3
```

### 7.2.6 Displays of Ownership and Protection

You can display ownership and protection information with the following commands and qualifiers:

Command	Display
DIRECTORY/ACL file-spec	File's ACL
DIRECTORY/OWNER_UIC file-spec	File's UIC
DIRECTORY/PROTECTION file-spec	File's UIC-based protection
DIRECTORY/SECURITY	All of the above
DIRECTORY/FULL file-spec	All of the above
SHOW ACL	Device, file, logical name table, or global section's ACL
SHOW PROCESS/ALL	Process UIC
SHOW PROTECTION	Default file protection
SHOW DEVICES/FULL device-name	Device UIC and protection

## 7.3 Creating and Deleting ACLs

Typically, you use the ACL editor to create an ACL or to make major changes to it, and you use the DCL commands SET ACL, SET FILE/ACL, SET DIRECTORY/ACL, and SET DEVICE/ACL to make minor changes to an ACL or to set an ACL on more than one object.

### 7.3.1 Using the SET ACL Command

The DCL command SET ACL allows you to specify an ACL for a file, directory, device, system logical name table, or global section. To specify an ACL for a device, for example, specify the SET ACL command with the /OBJECT\_TYPE=DEVICE qualifier. For example, the following command specifies an ACL that grants users who hold the identifier PAYROLL read access to the disk device WORKDISK:

```
$ SET ACL/OBJECT_TYPE=DEVICE/ACL=(IDENTIFIER=PAYROLL,ACCESS=READ) -
_$ WORKDISK
```

To set the same ACL on all files with the file type DAT in the default directory, specify the following command:

```
$ SET ACL/OBJECT_TYPE=FILE/ACL=(IDENTIFIER=PAYROLL,ACCESS=READ) -
_$ *.DAT;*
```

To select a subset of the files specified by the wildcard character, use one or more of the following qualifiers:

Qualifier	Meaning
/BY_OWNER	Selects files with a specified UIC
/EXCLUDE	Selects all but the specified files
/BEFORE	Selects files created or modified before a specified time
/SINCE	Selects files created or modified since a specified time
/CREATED	Specifies that /BEFORE or /SINCE check for the creation date

You can also use the /CONFIRM qualifier with those of the preceding list to elicit a confirmation prompt for each file; the /CONFIRM qualifier allows you to specify whether the file is to be affected.

By default, the DCL commands that create ACLs add the entries in the order specified to the top of the ACL. (If the object does not have an ACL, an ACL is created.) Use the /AFTER qualifier to specify a position for the entry (other than the top of the ACL). For example, the following command places a new entry after an existing entry—the ACE specified with the /AFTER qualifier—in the access control list of the file DOGS83.DAT.

```
$ SET ACL/AFTER=(IDENTIFIER=[JONES],ACCESS=READ) -
_$ /ACL=(IDENTIFIER=SECRET,ACCESS=READ+EXECUTE) DOGS83.DAT
```

The /AFTER qualifier is especially useful because the ACEs are processed from first to last. For example, if users holding the identifier PAYROLL are denied access in an ACE that precedes one granting access to user FRED, and if FRED holds both identifiers, FRED will be denied access.



The SET ACL command also permits the following ACL operations:

- Delete an ACE—Use the /DELETE qualifier to delete ACEs (specified on the /ACL qualifier) from an object's ACL. If no value is specified with the /ACL qualifier, all the entries in the ACL of the specified objects are deleted (except those entries protected by the PROTECTED option). The following example deletes the ACE specified with the /ACL qualifier from the ACL of the latest version of each file with type FOR in the default directory.
 

```
$ SET ACL/OBJECT_TYPE=FILE -
_$ /ACL=(IDENTIFIER=PAYROLL,ACCESS=READ+WRITE+EXECUTE) -
_$ /DELETE *.FOR
```
- Replace an ACE—Use the /REPLACE qualifier to specify an ACE to replace the ACE specified by the /ACL qualifier. If no value is specified with the /ACL qualifier, the ACEs specified by the /REPLACE qualifier are added to the ACLs of the specified object(s). The following example changes the ACE specified with /ACL to the one specified by /REPLACE in the ACL of the device WORKDISK.
 

```
$ SET ACL/ACL=(IDENTIFIER=[200,200],ACCESS=READ) -
_$ /REPLACE=(IDENTIFIER=NONEXEC,ACCESS=NONE) -
_$ /OBJECT_TYPE=DEVICE WORKDISK
```
- Copy an ACL—Use the /LIKE qualifier to copy an ACL from one object to another. Specify the name of the object whose ACL is to be copied as the value of the /LIKE qualifier and specify the name of the object(s) to receive the ACL as the parameter of the SET ACL command. The following example copies the ACL from [USER]X.X to all versions of all files in the [USER] directory tree. The /LOG qualifier displays each file specification as the file is modified. (You might also wish to include the /CONFIRM qualifier, which issues a request for confirmation before each modification.)
 

```
$ SET ACL/LOG/LIKE=[USER.TESTS]X.X [USER...]*.*;*
```
- Make an ACL the default—Use the /DEFAULT qualifier to set the ACL on an existing file to be the same as a newly created file. Any ACL that already exists on the file is deleted.

### 7.3.2 ACL Editor

To invoke the ACL editor, type EDIT/ACL followed by the name of the object whose ACL you wish to edit and press RETURN.

```
$ EDIT/ACL PROTA.TXT
```

If the object has an ACL, the ACL appears on the screen. Otherwise, you are creating an ACL and will begin the editing session by entering an ACE. You enter ACL editor commands by using the keypad keys pictured in the following diagram.

PF1 GOLD	PF2 HELP HELPMFT	PF3 FNDNXT FIND	PF4 DEL ACE UND ACE
7 SEL FIELD ADV FIELD	8	9	 DEL W UND W
4 ADVANCE BOTTOM	5 BACKUP TOP	6	' DEL C UND C
1 WORD	2 EOL DEL EOL	3	ENTER  ENTER
0 OVER ACE INSERT	• ITEM		

ZK-1740-84

By default, the ACL editor assumes that the object whose ACL is being edited is a file. To specify an object other than a file, use the /OBJECT\_TYPE qualifier. For example, to edit the ACL of a device named TTA1:, you would enter the following command line:

**\$ EDIT/ACL/OBJECT\_TYPE=DEVICE TTA1:**

You can specify any one of the following objects with the /OBJECT\_TYPE qualifier:

FILE	Specifies that the object type is a file or a directory file
DEVICE	Specifies that the object type is a device
SYSTEM_GLOBAL_SECTION	Specifies that the object type is a system global section
GROUP_GLOBAL_SECTION	Specifies that the object type is a group global section
LOGICAL_NAME_TABLE	Specifies that the object type is a logical name table

To execute a command, press the key. To execute the lower command shown on a key in the diagram, press the GOLD key first and then press the key. You can



display information on each command by pressing the HELP key followed by the command. To display information on the proper format of an ACE, press GOLD and then HELP.

To exit from the editor, and create or modify the ACL, press CTRL/Z. To exit, without creating or modifying the ACL, press GOLD and CTRL/Z. To refresh the screen while in the editor (for example, after receiving a broadcast message), enter CTRL/W.

The ACL editor edits a copy of the ACL. If your ACL editing session is terminated by entering CTRL/Y or if the system fails, the existing ACL remains unchanged or, if no ACL exists, no ACL is created. To recover an ACL editing session interrupted by CTRL/Y or system failure, specify the /RECOVER qualifier of the EDIT/ACL command.

### 7.3.2.1 Using Prompts

By default, the ACL editor prompts for each ACE and provides values in the various fields within an ACE whenever possible. The /MODE qualifier controls the choice of mode. To disable prompting, specify /MODE=NOPROMPT.

The FIELD, ITEM, and ENTER commands on the keypad enable you to take full advantage of prompt mode in the ACL editor.

- **FIELD**—Completes the current ACE field and moves the cursor to the next ACE field or subfield, inserting text as needed. If the ACL editor is not in prompt mode, the ACL editor advances to the next field in the current existing ACE.
- **ITEM**—Selects the next item for the current ACE field. If the ACL editor is not in prompt mode, this key is ignored.
- **ENTER**—Indicates that the current ACE is complete and ready for parsing. This key terminates the insertion. You can press it while the cursor is located at any position within the ACE. (Pressing RETURN produces the same results.)

### 7.3.2.2 Moving the Cursor

To position the cursor within the current line, use the following commands:

Command	Action
EOL	Moves the cursor to the end of the line
WORD	Moves the cursor one word in the current direction
ADVANCE	Sets the cursor direction to forward
BACKUP	Sets the cursor direction to backward
LEFT ARROW	Moves the cursor left one character
RIGHT ARROW	Moves the cursor right one character

Command	Action
FIELD	Completes the current ACE field and moves the cursor to the next ACE field or subfield
ITEM	Selects the next item for the current ACE field

The following commands move the cursor to a different line of the ACL.

Command	Action
BOTTOM	Moves the cursor to the end of the ACL
TOP	Moves the cursor to the beginning of the ACL
UP ARROW	Moves the cursor up a line
DOWN ARROW	Moves the cursor down a line
OVER ACE	Moves the cursor to the next ACE

To locate a specified text string or locate the next occurrence of a previously specified text string use FIND and FIND NEXT.

If the command moves the cursor out of an unprocessed ACE, the editor attempts to process the current ACE. If the ACE is improperly formatted, an error occurs and the cursor remains in place.

### 7.3.2.3 Entering and Deleting Data

Unless you disable prompting, the following text appears on the screen when you invoke the ACL editor.

(IDENTIFIER=

If you want to create an identifier ACE, specify the identifier value by typing in the appropriate value. Or you can delete the text and type a default\_protection or alarm\_journal ACE.

To delete text, use the following commands:

Command	Action
DEL C	Deletes the current character
DELETE key	Deletes the previous character
DEL W	Deletes the current word
LINE FEED	Deletes the previous word
DEL EOL	Deletes from cursor to end of line
DEL ACE	Deletes the current ACE
CTRL/U	Deletes the text from the cursor to the beginning of the line

If you delete text by mistake, you can restore it by using the following commands:

Command	Action
UND C	Restores the most recently deleted character
UND W	Restores the most recently deleted word
UND ACE	Restores the most recently deleted ACE
GOLD CTRL/U	Restores the most recently deleted portion of a line

#### 7.3.2.4 Processing an ACE

To process an ACE, press ENTER or carriage return. You must process the current ACE before you can begin editing a second ACE. If the ACE is in the proper format, the editor puts the ACE into the ACL and moves the cursor to the next line. If the cursor is positioned at the end of the ACL and if prompting is enabled, the editor displays the text IDENTIFIER=. (Moving the cursor off a line containing an unprocessed ACE implicitly processes the ACE.)

# Appendix ACL

## Access Control Lists

An access control list (ACL) specifies protection information for an object, such as a file, directory, device, system logical name table, or global section.

### ACL.1 Format of Access Control Entries

An access control entry has the following general format (parentheses and commas are required).

(type, option, access)

The only portion that is always required is the *type* field. The *option* and *access* fields may be optional, depending on the type of ACE being specified.

#### PARAMETERS

##### type

Indicates the kind of information to be supplied by the access control entry. Specify *type* as one of the following (abbreviations are allowed).

ALARM_JOURNAL=name	The name of an alarm journal to which security alarms are sent. The only name currently allowed is SECURITY. See the SET AUDIT command in Appendix DCL and Section 7.2.2.7 for more information about security auditing.
DEFAULT_PROTECTION	Default file protection information for a directory. Applicable only to an access control list of a directory file.
IDENTIFIER=identifier	Access information for an object (file, directory, or device). <i>Identifier</i> specifies the user(s) to whom the access information applies.

## ACL-2    Access Control Lists

### Format of Access Control Entries

#### options

Specifies whether the access control entry is propagated and whether it can be displayed and/or deleted. Specify *option* in the following format:

OPTIONS=keyword+...

Specify *keyword* as one of the following (may be abbreviated).

NONE (default)	No options.
DEFAULT	The access control entry is to be placed in the access control list of each file subsequently created in the directory. (The DEFAULT option is not copied with the rest of the access control entry.) Applicable only to an access control list of a directory file.
HIDDEN	The access control entry can be changed only by the program that created it. The HIDDEN option can only be specified from a program. (You cannot specify it using DCL commands.)
PROTECTED	The access control entry is not deleted by explicit (/DELETE) or implicit (/LIKE) attempts to delete the entire ACL. An access control entry marked as protected must be explicitly identified to be deleted (for example, /ACL=(ACE)/DELETE).
NOPROPAGATE	The ACE is not propagated when the ACL is copied from one version of a file to a later version of the file.

#### access

Specifies the type of access to be granted. If the access control entry is of IDENTIFIER type, specify *access* in the form

ACCESS=keyword+...

Specify *keyword* as one of the following (may be abbreviated).

NONE  
READ  
WRITE  
EXECUTE  
DELETE  
CONTROL

You may also specify the keywords SUCCESS and FAILURE with the ALARM\_JOURNAL ACE type.



## Access Control Lists    ACL-3

### Format of Access Control Entries

When applied to devices, access types have the following meanings:

NONE	No access to the device
READ	The right to allocate and issue read requests to the device
WRITE	The right to issue write requests to the device
CONTROL	The right to change the access control list
SUCCESS	Causes an alarm if access is granted to the device
FAILURE	Causes an alarm if access is denied to the device

When applied to files, access types have the following meanings:

NONE	No access to the file
READ	The right to read the file
WRITE	The right to read and write to the file
EXECUTE	The right to execute the file
DELETE	The right to delete the file
CONTROL	The right to change the file header; this implies that users with CONTROL access can give themselves any other type of access
SUCCESS	Causes an alarm if access is granted to the file
FAILURE	Causes an alarm if access is denied to the file

When applied to global sections, access types have the following meanings:

NONE	No access to the global section
READ	The right to map the section for read access
WRITE	The right to map the section for write access
EXECUTE	The right to map the section for execute access (available only to privileged software)
CONTROL	The right to change the access control list (applies only to page frame number (PFN) and page file global sections)
SUCCESS	Causes an alarm if access is granted to the global section
FAILURE	Causes an alarm if access is denied to the global section

## ACL-4 Access Control Lists

### Format of Access Control Entries

When applied to system logical name tables, access types have the following meanings:

NONE	No access to the logical name table
READ	The right to look up logical names in the table
WRITE	The right to create and delete logical names in the table
DELETE	The right to delete the logical name table
CONTROL	The right to change the access control list
SUCCESS	Causes an alarm if access is granted to the logical name table
FAILURE	Causes an alarm if access is denied to the logical name table

When applied to volumes, access types have the following meanings:

NONE	No access to the volume.
READ	The right to examine, print, or copy files on a volume (READ access on volumes limits the access to read only)
WRITE	The right to modify or to write existing files on a volume
EXECUTE	The right to create files on the volume and to write into them
DELETE	The right to delete files on the volume
CONTROL	The right to change the protection and ownership of the volume
SUCCESS	Causes an alarm if access is granted to the volume
FAILURE	Causes an alarm if access is denied to the volume

For a DEFAULT\_PROTECTION access control entry, specify *access* in the format *ownership:[access],...*

*Ownership* is the ownership category: SYSTEM, OWNER, GROUP, or WORLD. Each category may be abbreviated to the first character. *Access* is the access category: R (read), W (write), E (execute), or D (delete). To deny access to an ownership category, omit the access character after the ownership.

## ACL.2 EDIT/ACL Command

Use the DCL command EDIT/ACL to invoke the ACL editor.

### EDIT/ACL object-spec

Invokes the ACL editor. Enter CTRL/Z to exit from the ACL editor and save your edits; enter GOLD + CTRL/Z to quit the editing session without saving the edits.

## PARAMETERS

### **object-spec**

Specification of the object whose access control list is being edited. Specify a directory file as a file specification with the file type DIR. The file must be a disk file on a Files-11 Structure Level 2 formatted volume. No wildcard characters are allowed.

## QUALIFIERS

### **/JOURNAL[=file-spec] (default)**

### **/NOJOURNAL**

Specifies whether or not the ACL editor keeps a journal file during an editing session and the specification of that journal file. *File-spec* defaults to the name of the file being edited and a file type of JOURNAL.

### **/MODE=option**

Specifies whether or not the ACL editor prompts for field values. Specify the option as PROMPT (default) or NOPROMPT.

### **/OBJECT=type**

Specifies the type of object whose ACL is being edited. Possible types are:

FILE (default)	Specifies that the object is a file.
DEVICE	Specifies that the object is a device.
SYSTEM_GLOBAL_SECTION	Specifies that the object type is a system global section.
GROUP_GLOBAL_SECTION	Specifies that the object type is a group global section.
LOGICAL_NAME_TABLE	Specifies that the object type is a system logical name table.

### **/RECOVER[=file-spec]**

### **/NORECOVER (default)**

Specifies whether or not a journal file is executed before the editing session begins. *File-spec* defaults to the name of the file being edited and a file type of JOURNAL.

## EXAMPLES

**\$ EDIT/ACL CHAP.TXT**

Invokes the ACL editor to edit the access control list associated with the highest version of the file CHAP.TXT. If CHAP.TXT does not exist, an error occurs. If CHAP.TXT has no access control list, one is created when the user enters CTRL/Z.

**\$ EDIT/RECOVER/ACL CHAP.TXT**

Invokes the ACL editor to recover edits made during a previously aborted editing session.

## ACL-6 Access Control Lists




### ACL Editor Commands

## ACL.3 ACL Editor Commands

Invoke the ACL editor (with the EDIT/ACL command) to use the following commands.

### ACL.3.1 Keypad Diagram

Use the keypad to enter keypad commands.

PF1 GOLD	PF2 HELP HELPMFT	PF3 FNDNXT FIND	PF4 DEL ACE UND ACE
7 SEL FIELD ADV FIELD	8	9	 DEL W UND W
4 ADVANCE BOTTOM	5 BACKUP TOP	6	 DEL C UND C
1 WORD	2 EOL DEL EOL	3	ENTER  ENTER
0 OVER ACE INSERT	 ITEM		

ZK-1740-84

### ACL.3.2 Keypad Commands

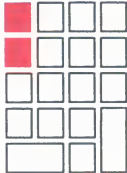
The diagram above the command description represents the keypad shown in Section ACL.3.1.

#### ADVANCE



Sets the cursor direction forward, from top to bottom of the access control list, for the commands OVER ACE, FIND, FNDNXT, MOVE SCREEN, and WORD, and remains in effect until you press BACKUP.

#### ADV FIELD



Completes the current ACE field and moves the cursor to the next ACE field, inserting text as needed.

#### BACKUP



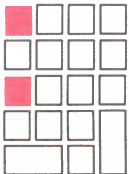
Sets cursor direction backward, from bottom to top of the access control list, for the commands OVER ACE, FIND, FNDNXT, MOVE SCREEN, and WORD, and remains in effect until you press ADVANCE.



## ACL-8 Access Control Lists

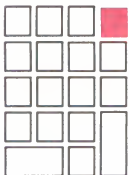
### ACL Editor Commands

#### BOTTOM



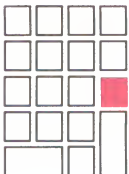
Moves the cursor to the end, or bottom, of the access control list. If prompting is enabled, begins the prompting sequence for a new ACE.

#### DEL ACE



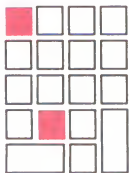
Deletes the current ACE (which may be more than 1 line).

#### DEL C



Deletes the character the cursor is on.

#### DEL EOL



Deletes the text from the cursor to the end of the line. The deleted text is stored in a buffer.

**DEL W**


Deletes text up to the first character of the next word.

**ENTER**


Completes and checks the syntax of the current ACE. The object's ACL is not modified at this time.

**EOL**


Moves the cursor to the end of the current line.

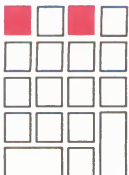
**FIELD**


Completes the current ACE field and moves the cursor to the next ACE field or subfield, inserting text as needed.

## ACL-10 Access Control Lists

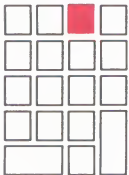
### ACL Editor Commands

#### FIND



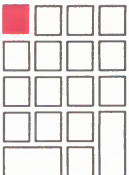
Displays the "Search for:" prompt and finds the first occurrence of the character string that you specify in response. You must terminate the string by pressing ENTER. If the string is found, the cursor moves to the first character in the string; otherwise, the cursor remains in place and the message "String was not found" appears.

#### FNDNXT



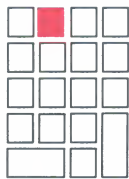
Moves the cursor to the first character of the next occurrence of the search string specified in the FIND command. If there is no further occurrence of the string, the cursor remains in place and the message "String was not found" appears.

#### GOLD



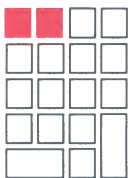
When pressed before another keypad key, specifies the second key's alternate function (the bottom function on the keypad diagram).

HELP



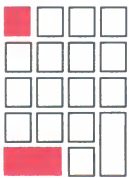
Displays a diagram of keypad keys. When one of the keys is pressed after HELP, information about that key is displayed. Pressing the TAB key after HELP is like pressing HELP FMT.

HELP FMT



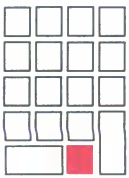
Displays the format of an ACE. By pressing the TAB key after HELP FMT, you can obtain information about each field and its possible values.

INSERT



Inserts a blank line before the current ACE. If prompting is enabled, begins the prompting sequence for a new ACE.

ITEM

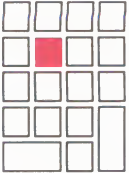


Selects the next value of the current ACE field.

## ACL-12 Access Control Lists

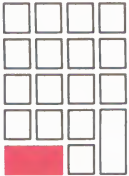
### ACL Editor Commands

#### MOVE SCREEN



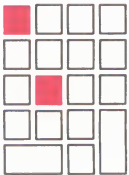
Moves the cursor one screen in the current direction (see ADVANCE or BACKUP). A screen is defined as two-thirds the number of lines in the display.

#### OVER ACE



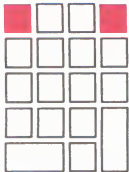
Moves the cursor to the beginning of the next ACE or to the beginning of the previous ACE.

#### TOP



Moves the cursor to the beginning, or top, of the access control list.

#### UND ACE



Inserts, above the current access control entry, the access control entry most recently deleted with DEL ACE.



**UND C**



Inserts to the left of the cursor the character most recently deleted with DEL C or DELETE.

**UND W**



Inserts to the left of the cursor the word most recently deleted with DELW or LINE FEED.

**WORD**



Moves the cursor one word in the current direction (see ADVANCE or BACKUP) within the current line.

Four arrow keys also move the cursor.

- **DOWN ARROW**—Moves the cursor to the character in the line below. If the access control entry in which the cursor is positioned is new, the ACL editor processes the access control entry before moving the cursor. If the entry is incomplete or improperly formatted, an error occurs and the cursor remains in place.
- **LEFT ARROW**—Moves the cursor one character to the left. If the cursor is at the left margin, LEFT ARROW moves the cursor to the rightmost character in the line above.

## ACL-14 Access Control Lists

### ACL Editor Commands

- RIGHT ARROW—Moves the cursor one character to the right. If the cursor is at the right margin, RIGHT ARROW moves it to the leftmost character in the line below.
- UP ARROW—Moves the cursor to the character in the line above it. If the access control entry in which the cursor is positioned is new, the ACL editor processes the access control entry before moving the cursor. If the entry is incomplete or improperly formatted, an error occurs and the cursor remains in place.

You can use the following keyboard keys to supplement the keypad keys. Keys in parentheses are for a VT100 series terminal.

- F12 (BACKSPACE)—Moves the cursor to the beginning of the current line.
- <X> (DELETE)—Deletes the character to the left of the cursor.
- F13 (LINE FEED)—Deletes the text from the cursor to the beginning of the word. If cursor is positioned at the first character of the word, deletes to the beginning of the previous word.
- TAB (TAB)—Moves the text to the next tab stop.

On the VT200 series terminal, you can also use the following supplemental editing keypad keys:

- FIND (E1)—Elicits the *Search for:* prompt as the first step in the FIND operation. Type the search string after the prompt and then press either the DO or ENTER key to process the search. (Performs the same function as the FIND keypad key.)
- INSERT HERE (E2)—Indicates the line where the selected text in the PASTE buffer is to be inserted. By default, support for the PASTE buffer is disabled.
- REMOVE (E3)—Removes the text within the select range to the PASTE buffer. Each time REMOVE is used, the previous contents of the PASTE buffer are deleted. By default, support for the PASTE buffer is disabled.
- COPY (GOLD E3)—Copies the text within the select range to the PASTE buffer. Each time COPY is used, the previous contents of the PASTE buffer are deleted. By default, support for the PASTE buffer is disabled.
- SELECT (E4)—Marks the beginning of a range of text to be removed or copied to the PASTE buffer. Press SELECT; move the cursor to include the desired amount of text to be removed or copied; and press either REMOVE (E3) or COPY (GOLD E3) to complete the operation. By default, support for the PASTE buffer is disabled.
- PREV SCREEN (E5)—Moves the cursor one screen in the backward direction. By default, a screen is defined as two-thirds the number of lines in the display.

- NEXT SCREEN (**E6**)—Moves the cursor one screen in the forward direction. By default, a screen is defined as two-thirds the number of lines in the display.

### **ACL.3.3 Control Keys**

The following control keys also perform editing functions:

- CTRL/A—Determines whether characters are entered in insert or overstrike mode. Insert mode (the default) inserts a character to the left of the current character. Overstrike mode replaces the current character.
- CTRL/D—Allows you to execute one TPU command.
- CTRL/H—Moves the cursor to the beginning of the current line.
- CTRL/J—Deletes the text from the cursor to the beginning of the word. If the cursor is positioned at the first character of the word, CTRL/J deletes to the beginning of the previous word.
- CTRL/U—Deletes the text from the cursor to the beginning of the line.
- CTRL/W—Refreshes the screen by deleting extraneous characters and restoring the previous display.
- CTRL/Z—Terminates the editing session and updates the ACL.
- GOLD + CTRL/Z—Ignores changes made to the ACL and terminates the editing session.

## **ACL.4 DCL Commands That Affect ACLs**

Use the following DCL commands to create, modify, or delete a single access control entry or to manipulate an entire access control list.

### **SET ACL object**

Creates or modifies one or more access control entries (ACEs) in the access control list (ACL) of the specified object.

#### **PARAMETERS**

##### **object**

The specification of an object whose access control list is to be modified. Files must be on Files-11 Structure Level 2 formatted disk volumes. Logical name tables must be system logical name tables.

## ACL-16 Access Control Lists

### DCL Commands That Affect ACLs

#### QUALIFIERS

##### **/ACL[=(ace,...)]**

Specifies the access control list or entries to be created or modified. *Ace* specifies an access control to be inserted at the top of the ACL. If no ACE is specified, the entire ACL is affected. (Note that security alarm ACEs are always placed at the beginning of the ACL.)

##### **/AFTER=ace**

Places the access control entries specified with the */ACL* qualifier after the access control entry specified with */AFTER*. By default, access control entries are added to the beginning of the access control list. (Note that security alarm ACEs are always placed at the beginning of the ACL.)

##### **/BEFORE[=time]**

Modifies the ACLs of only those files dated before the specified time. You can specify time as an absolute time, a combination of absolute and delta times, or one of the following keywords: TODAY (default), TOMORROW, YESTERDAY. Compatible only with */OBJECT\_TYPE=FILE*.

##### **/BY\_OWNER[=uic]**

Modifies only the ACLs of those files with the specified user identification code (UIC). The default UIC is that of the current process. Compatible only with */OBJECT\_TYPE=FILE*.

##### **/CONFIRM**

##### **/NOCONFIRM (default)**

Issues a request for confirmation before each modification. Compatible only with */OBJECT\_TYPE=FILE*. The following responses are valid:

YES	Modify the ACE.
NO	Do not modify the ACE.
TRUE	Modify the ACE.
FALSE	Do not modify the ACE.
1	Modify the ACE.
0	Do not modify the ACE.
RETURN	Do not modify the ACE.
ALL	Continue execution of the command with no further confirmation prompts.
CTRL/Z	Stop execution of the command.
QUIT	Stop execution of the command.

**/CREATED**

Modifies the ACLs of files selected according to their creation date. Relevant only with the /BEFORE and /SINCE qualifiers. Compatible only with /OBJECT\_TYPE=FILE.

**/DEFAULT**

Creates a new ACL using the default ACEs of the parent (not the default) directory. For a directory file, the /DEFAULT qualifier propagates the entire ACL (except for ACEs with the NOPROPAGATE options) so that a particular access protection can be propagated throughout a directory tree. For all other files, the /DEFAULT qualifier propagates the DEFAULT option ACEs in the ACL of the parent directory to the ACL of the specified files. Compatible only with /OBJECT\_TYPE=FILE.

**/DELETE**

Deletes the access control entries specified with the /ACL qualifier. If no access control entry is specified with /ACL, the entire access control list is deleted (excluding ACEs specified with the PROTECTED option).

**/EXCLUDE=(file-spec,...)**

Excludes the specified files from the operation. Wildcard characters are allowed. However, you cannot use relative version numbers to exclude a specific version. Compatible only with /OBJECT\_TYPE=FILE.

**/LIKE=(OBJECT\_TYPE=type,OBJECT\_NAME=name)**

Deletes the access control list of the specified object and replaces it with the access control list of the object specified with /LIKE. (The source and destination objects do not have to be the same type for this qualifier.) You can specify the following keywords for /OBJECT\_TYPE: DEVICE, FILE, SYSTEM\_GLOBAL\_SECTION, GROUP\_GLOBAL\_SECTION, or LOGICAL\_NAME\_TABLE.

**/LOG**

**/NOLOG (default)**

Displays the specification of each object whose ACL is modified as the command executes.

**/NEW**

Deletes the ACL (except ACEs with the PROTECTED option) of the specified object and replaces it with the ACL specified with the /ACL or /LIKE qualifier.

**/OBJECT\_TYPE=(keyword)**

Specifies the type of the object whose ACL is being modified. Possible keywords are: FILE (default), DEVICE, SYSTEM\_GLOBAL\_SECTION, GROUP\_GLOBAL\_SECTION, or LOGICAL\_NAME\_TABLE.



## ACL-18 Access Control Lists

### DCL Commands That Affect ACLs

#### **/REPLACE=(ace,...)**

Deletes the access control entries specified with the /ACL qualifier and replaces them with the access control entries specified with /REPLACE.

#### **/SINCE[=time]**

Modifies only the ACLs of those files dated after the specified time. You can specify time as an absolute time, a combination of absolute and delta times, or one of the following keywords: TODAY (default), TOMORROW, and YESTERDAY. Compatible only with /OBJECT\_TYPE=FILE.

#### EXAMPLES

```
$ SET ACL/ACL=(IDENTIFIER=[SMITH],ACCESS=CONTROL) *.FOR
```

Adds the access control entry specified by /ACL to the beginning of the access control list of each file with a file type of FOR in the current default directory. (The /OBJECT\_TYPE qualifier defaults to file.)

```
$ SET ACL/ACL=(IDENTIFIER=NONEEXEC,ACCESS=READ) -  
_$ /REPLACE=(IDENTIFIER=NONEEXEC,ACCESS=NONE) -  
_$ /OBJECT_TYPE=DEVICE WORKDISK
```

Changes the access control entry specified by /ACL to the one specified by /REPLACE in the access control list of the device WORKDISK.

#### **SET DEVICE/ACL [(ace,...)] device-name,...**

Creates or modifies one or more access control entries in the access control lists of the specified devices.

#### PARAMETERS

##### **ace**

Access control entries (ACEs) to be modified. If no ACE is specified, the entire access control list (ACL) is affected.

##### **device-name**

Specification of devices whose access control lists are being edited; no wildcard characters are allowed.

#### QUALIFIERS

##### **/AFTER=ace**

Places the access control entries specified with the /ACL qualifier after the access control entry specified with /AFTER. By default, access control entries are added to the beginning of the access control list. (Note that security alarm ACEs are always placed at the beginning of the ACL.)



**/DELETE**

Deletes the access control entries specified with the /ACL qualifier. If no access control entry is specified with /ACL, the entire access control list is deleted (except ACEs specified with the PROTECTED option).

**/LIKE=device-name**

Deletes the access control list of the specified device and replaces it with the access control list of the file specified with /LIKE. No wildcard characters are allowed.

**/LOG**

**/NOLOG (default)**

Displays the device name of each device whose ACL is modified as the command executes.

**/NEW**

Deletes the access control lists (except ACEs with the PROTECTED option) of the specified devices and replaces them with the access control list specified with the /ACL or /LIKE qualifier.

**/REPLACE=(ace,...)**

Deletes the access control entries specified with the /ACL qualifier and replaces them with the access control entries specified with /REPLACE.

**EXAMPLES**

**\$ SET DEVICE/ACL=(IDENTIFIER=[FRED],ACCESS=NONE) WORKDISK**

Adds an access control entry specified by /ACL to the beginning of the access control list of the device WORKDISK.

**\$ SET DEVICE/ACL/LIKE=\$CONSOLE TERMINAL2**

Replaces the access control list of TERMINAL2 with the access control list of the console terminal (\$CONSOLE).

**SET DIRECTORY/ACL [(ace,...)] directory-spec,...**

Creates or modifies one or more access control entries (ACEs) in the access control lists (ACLs) of the specified directory.

**PARAMETERS**

**ace**

Access control entries (ACEs) to be modified. If no ACE is specified, the entire access control list (ACL) is affected.

## **ACL-20    Access Control Lists**

### **DCL Commands That Affect ACLs**

#### **directory-spec**

Specifications of directory whose access control list is being edited. Device name and colon are optional. Wildcard characters are allowed. Directories must be on a Files-11 Structure Level 2 formatted disk volume.

#### **QUALIFIERS**

##### **/AFTER=ace**

Places the access control entries specified with the /ACL qualifier after the access control entry specified with /AFTER. By default, access control entries are added to the beginning of the access control list. (Note that security alarm ACEs are always placed at the beginning of the ACL.)

##### **/BEFORE[=time]**

Modifies only the ACLs of those directories dated before the specified time. You can specify time as an absolute time, a combination of absolute and delta times, or one of the following keywords: TODAY (default), TOMORROW, YESTERDAY.

##### **/BY\_OWNER[=uic]**

Modifies only the ACLs of those directories with the specified user identification code (UIC). The default UIC is that of the current process.

##### **/CONFIRM**

##### **/NOCONFIRM (default)**

Issues a request for confirmation before each modification. The following responses are valid:

YES	Modify the ACE.
NO	Do not modify the ACE.
TRUE	Modify the ACE.
FALSE	Do not modify the ACE.
1	Modify the ACE.
0	Do not modify the ACE.
RETURN	Do not modify the ACE.
ALL	Continue execution of the command with no further confirmation prompts.
CTRL/Z	Stop execution of the command.
QUIT	Stop execution of the command.

##### **/CREATED**

Modifies the ACLs of directories selected according to their creation date. Relevant only with the /BEFORE and /SINCE qualifiers.

**/DEFAULT**

Creates a new ACL using the default ACEs of the parent (not default) directory. For a directory file, the /DEFAULT qualifier propagates the entire ACL (except for ACEs with the NOPROPAGATE option) so that a particular access protection can be propagated throughout a directory tree.

**/DELETE**

Deletes the access control entries specified with the /ACL qualifier. If no access control entry is specified with /ACL, the entire access control list is deleted (except ACEs specified with the PROTECTED option).

**/EXCLUDE=(directory-spec,...)**

Excludes the specified directory from the operation. Wildcard characters are allowed.

**/LIKE=directory-spec**

Deletes the access control list of the specified directory and replaces it with the access control list of the directory specified with /LIKE. Wildcard characters are not allowed.

**/LOG**

**/NOLOG (default)**

Displays the specification of each directory modified as the command executes.

**/NEW**

Deletes the access control list (except ACEs set with the PROTECTED option) of the specified directory and replaces it with the access control entries specified with the /ACL qualifier.

**/REPLACE=(ace,...)**

Deletes the access control entries specified with the /ACL qualifier and replaces them with the access control entries specified with /REPLACE.

**/SINCE[=time]**

Modifies only those directories dated after the specified time. You can specify time as an absolute time, a combination of absolute and delta times, or one of the following keywords: TODAY (default), TOMORROW, YESTERDAY.

**EXAMPLE**

```
$ SET DIRECTORY/ACL/LIKE=[SMITH.PERSONAL] [JONES.PERSONAL]
```

Replaces the ACL of the directory [JONES.PERSONAL] with the ACL of the directory [SMITH.PERSONAL].

```
$ SET DIRECTORY/ACL=(IDENTIFIER=[123,321]+NETWORK,ACCESS=NONE)
```

Adds an identifier that permits no network access to the directory for user [123,321].

## **ACL-22    Access Control Lists**

### **DCL Commands That Affect ACLs**

#### **SET FILE/ACL [= (ace,...)] file-spec,...**

Modifies one or more access control entries (ACEs) in the access control lists (ACLs) of the specified files.

#### **PARAMETERS**

##### **ace**

Access control entries (ACEs) to be modified. If no ACE is specified, the entire access control list (ACL) is affected.

##### **file-spec**

Specifications of files whose access control lists are being edited. Wildcard characters are allowed. Use a comma to separate file specifications. Each file must be on a Files-11 Structure Level 2 formatted disk volume.

#### **QUALIFIERS**

##### **/AFTER=ace**

Places the access control entries specified with the /ACL qualifier after the access control entry specified with /AFTER. By default, access control entries are added to the beginning of the access control list. (Note that security alarm ACEs are always placed at the beginning of the ACL.)

##### **/BEFORE[=time]**

Modifies the ACLs of only those files dated before the specified time. You can specify time as an absolute time, a combination of absolute and delta times, or one of the following keywords: TODAY (default), TOMORROW, and YESTERDAY.

##### **/BY\_OWNER[=uic]**

Modifies the ACLs of only those files with the specified user identification code. The default UIC is that of the current process.

##### **/CONFIRM**

##### **/NOCONFIRM (default)**

Issues a request for confirmation before each modification. The following responses are valid:

YES	Modify the ACE.
NO	Do not modify the ACE.
TRUE	Modify the ACE.
FALSE	Do not modify the ACE.
1	Modify the ACE.
0	Do not modify the ACE.

## Access Control Lists **ACL-23**

### DCL Commands That Affect ACLs

RETURN	Do not modify the ACE.
ALL	Continue execution of the command with no further confirmation prompts.
CTRL/Z	Stop execution of the command.
QUIT	Stop execution of the command.

#### **/CREATED**

Modifies the ACLs of files selected according to their creation date. Relevant only with the /BEFORE and /SINCE qualifiers.

#### **/DEFAULT**

Creates a new ACL using the default ACEs of the parent (not default) directory. For a directory file, the /DEFAULT qualifier propagates the entire ACL (except for ACEs with the NOPROPAGATE option) so that a particular access protection can be propagated throughout a directory tree. For all other files, the /DEFAULT qualifier propagates the DEFAULT option ACEs in the ACL of the parent directory to the ACL of the specified files.

#### **/DELETE**

Deletes the access control entries specified with the /ACL qualifier. If no access control entry is specified with /ACL, the entire access control list is deleted (except ACEs specified with the PROTECTED option).

#### **/EXCLUDE=(file-spec,...)**

Excludes the specified files from the operation. You can specify a directory name but not a device name in the file specifications. Wildcard characters are allowed. However, you cannot use relative version numbers to exclude a specific version.

#### **/LIKE=file-spec**

Deletes the access control list of the input files and replaces it with the access control list of the file specified with /LIKE.

#### **/LOG**

#### **/NOLOG (default)**

Displays the file specification of each file modified as the command executes.

#### **/NEW**

Deletes the access control list of the input file (except those entries set with the PROTECTED option) and replaces it with the access control list specified with the /ACL or /LIKE qualifier.

## ACL-24 Access Control Lists

### DCL Commands That Affect ACLs

#### **/REPLACE=(ace,...)**

Deletes the access control entries specified with the /ACL qualifier and replaces them with the access control entries specified with /REPLACE.

#### **/SINCE[=time]**

Modifies only those input files dated after the specified time. You can specify time as an absolute time, a combination of absolute and delta times, or one of the following keywords: TODAY (default), TOMORROW, YESTERDAY.

#### EXAMPLES

```
$ SET FILE/ACL=(IDENTIFIER=[200,200],ACCESS=R+W+E+D) *.*;*
```

Adds the access control entry specified by /ACL to the beginning of the access control list of each file in the current default directory.

```
$ SET FILE/ACL=(IDENTIFIER=[200,200],ACCESS=R+W+E+D)/EXCLUDE=*.DAT *.*;*
```

Adds the access control entry specified by /ACL to the beginning of the access control list of each file in the current default directory, except those files with a file type of DAT.

```
$ SET FILE/ACL/LIKE=[USER]PROTO.TXT [USER...] *.*;*
```

Deletes the access control lists from all files in the [USER] directory tree and replaces them with the access control list of the file [USER]PROTO.TXT.

```
$ SET FILE/ACL=(IDENTIFIER=[200,200],ACCESS=R+W+E+D) -  
_$ /REPLACE=(IDENTIFIER=[200,200],ACCESS=NONE) *.*;*
```

Changes the access control entry specified by /ACL to the one specified by /REPLACE in the access control list of each file in the current default directory.

```
$ SET FILE/ACL=(IDENTIFIER=[200,200],ACCESS=R+W+E+D)/NEW *.*;*
```

Replaces the access control list of each file in the current default directory with an access control list containing the access control entry specified by /ACL.

## **SHOW ACL object-name**

Displays the access control list (ACL) of an object.

#### PARAMETERS

##### **object-name**

The name of the object whose ACL is to be displayed. No wildcards are allowed.



## QUALIFIERS

### **/OBJECT\_TYPE=type**

Specifies the type of object whose ACL is to be displayed. Possible keywords are: FILE (the default), DEVICE, SYSTEM\_GLOBAL\_SECTION, GROUP\_GLOBAL\_SECTION, or LOGICAL\_NAME\_TABLE.

## EXAMPLE

**\$ SHOW ACL/OBJECT\_TYPE=DEVICE TTA1**

Displays the ACL of the device TTA1.



# Appendix CHAR

## Character Sets

The following tables present the ASCII character set and the DEC Multinational Character Set.

### CHAR.1 ASCII Character Set

The ASCII character set consists of the characters shown in the following table. The characters with names are not printable. (The ASCII character set comprises the first 127 characters of the DEC Multinational Character Set; descriptions of the nonprintable characters are located in the table in Section CHAR.3.) You can calculate the numeric value of a character by constructing a 2-digit hexadecimal number in which the column position of the character represents the 16s position of the hexadecimal number and the row position of the character represents the units position of the number. For example, an uppercase A has the numeric value 41 hexadecimal. String comparisons are made using these values.

**CHAR-2    Character Sets**  
**ASCII Character Set**

PF1  GOLD	PF2 HELP  DIR/FOLDER	PF3 EXTRACT/MAIL  EXTRACT	PF4 ERASE  SELECT/MAIL
7  SEND  SEND/EDIT	8  REPLY  REP/EDIT/EXT	9  FORWARD  FORWARD/EDIT	— READ/NEW  SHOW/NEW
4  CURRENT  CURRENT/EDIT	5  FIRST  FIRST/EDIT	6  LAST  LAST/EDIT	, DIR/NEW  DIR MAIL
1  BACK  BACK/EDIT	2  PRINT  PRINT/PR/NOT	3  DIR  DIR/ST=99999	ENTER   SELECT
0  NEXT  NEXT/EDIT	•  FILE  DELETE		

ZK 1744-84

**CHAR.2    ASCII and DEC Multinational Character Set Tables**

The table below represents the ASCII character set (characters with decimal values 0 through 127). The first half of each of the numbered columns identifies the character as you would enter it on a VT200 or VT100 series terminal or as you would see it on a printer (except for the nonprintable characters). The remaining half of each column identifies the character by the binary value of the byte; the value is stated in three radices—octal, decimal, and hexadecimal. For example, the letter uppercase A has, under ASCII conventions, a storage value of hexadecimal 41 (a bit configuration of 01000001), equivalent to 101 in octal notation and 65 in decimal notation.

# Character Sets CHAR-3

## ASCII and DEC Multinational Character Set Tables

COLUMN		0		1		2		3		4		5		6		7	
BITS		0 0 0 0		0 0 0 1		0 0 1 0		0 0 1 1		0 1 0 0		0 1 0 1		0 1 1 0		0 1 1 1	
ROW	b8 b7 b6 b5 b4 b3 b2 b1																
0	0 0 0 0	NUL	0 0 0 0	DLE	20 16 10 10	SP	40 32 20 20	0	60 48 30 30	@	100 64 40 40	P	120 80 50 50	`	140 96 60 60	p	160 112 70 70
1	0 0 0 1	SOH	1 1 1 1	DC1 (XON)	21 17 11 11	!	41 33 21 21	1	61 49 31 31	A	101 65 41 41	Q	121 81 51 51	a	141 97 61 61	q	161 113 71 71
2	0 0 1 0	STX	2 2 2 2	DC2	22 18 12 12	"	42 34 22 22	2	62 50 32 32	B	102 66 42 42	R	122 82 52 52	b	142 98 62 62	r	162 114 72 72
3	0 0 1 1	ETX	3 3 3 3	DC3 (XOFF)	23 19 13 13	#	43 35 23 23	3	63 51 33 33	C	103 67 43 43	S	123 83 53 53	c	143 99 63 63	s	163 115 73 73
4	0 1 0 0	EOT	4 4 4 4	DC4	24 20 14 14	\$	44 36 24 24	4	64 52 34 34	D	104 68 44 44	T	124 84 54 54	d	144 100 64 64	t	164 116 74 74
5	0 1 0 1	ENQ	5 5 5 5	NAK	25 21 15 15	%	45 37 25 25	5	65 53 35 35	E	105 69 45 45	U	125 85 55 55	e	145 101 65 65	u	165 117 75 75
6	0 1 1 0	ACK	6 6 6 6	SYN	26 22 16 16	&	46 38 26 26	6	66 54 36 36	F	106 70 46 46	V	126 86 56 56	f	146 102 66 66	v	166 118 76 76
7	0 1 1 1	BEL	7 7 7 7	ETB	27 23 17 17	'	47 39 27 27	7	67 55 37 37	G	107 71 47 47	W	127 87 57 57	g	147 103 67 67	w	167 119 77 77
8	1 0 0 0	BS	10 8 8 8	CAN	30 24 18 18	(	50 40 28 28	8	70 56 38 38	H	110 72 48 48	X	130 88 58 58	h	150 104 68 68	x	170 120 78 78
9	1 0 0 1	HT	11 9 9 9	EM	31 25 19 19	)	51 41 29 29	9	71 57 39 39	I	111 73 49 49	Y	131 89 59 59	i	151 105 69 69	y	171 121 79 79
10	1 0 1 0	LF	12 10 A A	SUB	32 26 1A 1A	*	52 42 2A 2A	:	72 58 3A 3A	J	112 74 4A 4A	Z	132 90 5A 5A	j	152 106 6A 6A	z	172 122 7A 7A
11	1 0 1 1	VT	13 11 B B	ESC	33 27 1B 1B	+	53 43 2B 2B	;	73 59 3B 3B	K	113 75 4B 4B	[	133 91 5B 5B	k	153 107 6B 6B	{	173 123 7B 7B
12	1 1 0 0	FF	14 12 C C	FS	34 28 1C 1C	,	54 44 2C 2C	<	74 60 3C 3C	L	114 76 4C 4C	\	134 92 5C 5C	l	154 108 6C 6C		174 124 7C 7C
13	1 1 0 1	CR	15 13 D D	GS	35 29 1D 1D	-	55 45 2D 2D	=	75 61 3D 3D	M	115 77 4D 4D	]	135 93 5D 5D	m	155 109 6D 6D	}	175 125 7D 7D
14	1 1 1 0	SO	16 14 E E	RS	36 30 1E 1E	.	56 46 2E 2E	>	76 62 3E 3E	N	116 78 4E 4E	^	136 94 5E 5E	n	156 110 6E 6E	~	176 126 7E 7E
15	1 1 1 1	SI	17 15 F F	US	37 31 1F 1F	/	57 47 2F 2F	?	77 63 3F 3F	O	117 79 4F 4F	_	137 95 5F 5F	o	157 111 6F 6F	DEL	177 127 7F 7F

### KEY

CHARACTER	ESC	33 27 1B	OCTAL DECIMAL HEX
-----------	-----	----------------	-------------------------

ZK-1752-84

The ASCII character set comprises the first half of the DEC Multinational Character Set. The following table represents the second half of the DEC Multinational Character Set (characters with decimal values 128 through 255). The first half of each of the numbered columns identifies the character as you would see it on a

# CHAR-4 Character Sets

## ASCII and DEC Multinational Character Set Tables

VT200 series terminal or printer (these characters cannot be output on a VT100 series terminal). Section CHAR.3 describes how to enter symbols from the DEC Multinational Character Set.

8	9	10	11	12	13	14	15	COLUMN	ROW
1 0 0 0	1 0 0 1	1 0 1 0	1 0 1 1	1 1 0 0	1 1 0 1	1 1 1 0	1 1 1 1	b8 b7 b6 b5 b4 b3 b2 b1	
200 128 80	DCS	220 144 90	240 160 A0	260 176 B0	300 192 C0	320 208 D0	340 224 E0	360 240 F0	0 0 0 0
201 129 81	PU1	221 145 91	241 161 A1	261 177 B1	301 193 C1	321 209 D1	341 225 E1	361 241 F1	0 0 0 1
202 130 82	PU2	222 146 92	242 162 A2	262 178 B2	302 194 C2	322 210 D2	342 226 E2	362 242 F2	0 0 1 0
203 131 83	STS	223 147 93	243 163 A3	263 179 B3	303 195 C3	323 211 D3	343 227 E3	363 243 F3	0 0 1 1
204 132 84	CCH	224 148 94	244 164 A4	264 180 B4	304 196 C4	324 212 D4	344 228 E4	364 244 F4	0 1 0 0
205 133 85	NEL	225 149 95	245 165 A5	265 181 B5	305 197 C5	325 213 D5	345 229 E5	365 245 F5	0 1 0 1
206 134 86	SSA	226 150 96	246 166 A6	266 182 B6	306 198 C6	326 214 D6	346 230 E6	366 246 F6	0 1 1 0
207 135 87	ESA	227 151 97	247 167 A7	267 183 B7	307 199 C7	327 215 D7	347 231 E7	367 247 F7	0 1 1 1
210 136 88	HTS	230 152 98	250 168 A8	270 184 B8	310 200 C8	330 216 D8	350 232 E8	370 248 F8	1 0 0 0
211 137 89	HTJ	231 153 99	251 169 A9	271 185 B9	311 201 C9	331 217 D9	351 233 E9	371 249 F9	1 0 0 1
212 138 8A	VTS	232 154 9A	252 170 AA	272 186 BA	312 202 CA	332 218 DA	352 234 EA	372 250 FA	1 0 1 0
213 139 8B	PLD	233 155 9B	253 171 AB	273 187 BB	313 203 CB	333 219 DB	353 235 EB	373 251 FB	1 0 1 1
214 140 8C	PLU	234 156 9C	254 172 AC	274 188 BC	314 204 CC	334 220 DC	354 236 EC	374 252 FC	1 1 0 0
215 141 8D	RI	235 157 9D	255 173 AD	275 189 BD	315 205 CD	335 221 DD	355 237 ED	375 253 FD	1 1 0 1
216 142 8E	SS2	236 158 9E	256 174 AE	276 190 BE	316 206 CE	336 222 DE	356 238 EE	376 254 FE	1 1 1 0
217 143 8F	SS3	237 159 9F	257 175 AF	277 191 BF	317 207 CF	337 223 DF	357 239 EF	377 255 FF	1 1 1 1

### KEY

CHARACTER

ESC

33

OCTAL

27

DECIMAL

1B

HEX



### **CHAR.3    DEC Multinational Character Set**

The DEC Multinational Character Set is an 8-bit character set with 256 characters; the first 128 characters in the set correspond to the ASCII character set. Each character has a value in the range 0 through 255 decimal.

In the following table, the graphic symbols shown in parentheses represent ASCII control characters. These are produced on most terminals by pressing the key indicated while holding down the CONTROL key. On VT200 series terminals, graphic symbols with decimal values greater than 127 can be entered using the compose sequences. Press the Compose Character key followed by the EDT symbol; the graphic symbol is then displayed on your terminal. On VT200 series terminals, you can enter symbols for characters 128 through 255 either in EDT or at DCL level.

On VT100 series terminals, graphic symbols with decimal values greater than 127 can only be entered from screen mode in EDT. Use the EDT keypad command SPECINS or the nokeypad command ASC to enter these characters in your text; EDT then displays the EDT symbol that corresponds to the character rather than displaying the graphic symbol itself.

# CHAR-6 Character Sets

## DEC Multinational Character Set

Graphic	EDT Symbol	Decimal Value	Abbrev.	Description
(@)	^@	0	NUL	null character
(A)	^A	1	SOH	start of heading
(B)	^B	2	STX	start of text
(C)	^C	3	ETX	end of text
(D)	^D	4	EOT	end of transmission
(E)	^E	5	ENQ	enquiry
(F)	^F	6	ACK	acknowledge
(G)	^G	7	BEL	bell
(H)	^H	8	BS	backspace
(I)		9	HT	horizontal tabulation
(J)	<LF>	10	LF	line feed
(K)	<VT>	11	VT	vertical tabulation
(L)	<FF>	12	FF	form feed
(M)	<CR>	13	CR	carriage return
(N)	^N	14	SO	shift out
(O)	^O	15	SI	shift in
(P)	^P	16	DLE	data link escape
(Q)	^Q	17	DC1	device control 1
(R)	^R	18	DC2	device control 2
(S)	^S	19	DC3	device control 3
(T)	^T	20	DC4	device control 4
(U)	^U	21	NAK	negative acknowledge
(V)	^V	22	SYN	synchronous idle
(W)	^W	23	ETB	end of transmission block
(X)	^X	24	CAN	cancel
(Y)	^Y	25	EM	end of medium
(Z)	^Z	26	SUB	substitute
([)	<ESC>	27	ESC	escape
(\)	^\	28	FS	file separator
(])	^]	29	GS	group separator
(^)	^^	30	RS	record separator
(_)	^_	31	US	unit separator
		32	SP	space
!	!	33	!	exclamation point
"	"	34	"	quotation marks (double quote)
#	#	35	#	number sign
\$	\$	36	\$	dollar sign
%	%	37	%	percent sign
&	&	38	&	ampersand
'	'	39	'	apostrophe (single quote)
(	(	40	(	opening parenthesis

**Character Sets    CHAR-7**  
**DEC Multinational Character Set**

Graphic	EDT Symbol	Decimal Value	Abbrev.	Description
)	)	41	)	closing parenthesis
*	*	42	*	asterisk
+	+	43	+	plus
,	,	44	,	comma
-	-	45	-	hyphen or minus
.	.	46	.	period or decimal point
/	/	47	/	slash
0	0	48	0	zero
1	1	49	1	one
2	2	50	2	two
3	3	51	3	three
4	4	52	4	four
5	5	53	5	five
6	6	54	6	six
7	7	55	7	seven
8	8	56	8	eight
9	9	57	9	nine
:	:	58	:	colon
;	;	59	;	semicolon
<	<	60	<	less than
=	=	61	=	equals
>	>	62	>	greater than
?	?	63	?	question mark
@	@	64	@	commercial at
A	A	65	A	uppercase A
B	B	66	B	uppercase B
C	C	67	C	uppercase C
D	D	68	D	uppercase D
E	E	69	E	uppercase E
F	F	70	F	uppercase F
G	G	71	G	uppercase G
H	H	72	H	uppercase H
I	I	73	I	uppercase I
J	J	74	J	uppercase J
K	K	75	K	uppercase K
L	L	76	L	uppercase L
M	M	77	M	uppercase M
N	N	78	N	uppercase N
O	O	79	O	uppercase O
P	P	80	P	uppercase P

**CHAR-8 Character Sets**  
**DEC Multinational Character Set**

Graphic	EDT Symbol	Decimal Value	Abbrev.	Description
Q	Q	81	Q	uppercase Q
R	R	82	R	uppercase R
S	S	83	S	uppercase S
T	T	84	T	uppercase T
U	U	85	U	uppercase U
V	V	86	V	uppercase V
W	W	87	W	uppercase W
X	X	88	X	uppercase X
Y	Y	89	Y	uppercase Y
Z	Z	90	Z	uppercase Z
[	[	91	[	opening bracket
\	\	92	\	back slash
]	]	93	]	closing bracket
^	^	94	^	circumflex
_	_	95	_	underline (underscore)
`	`	96	`	grave accent
a	a	97	a	lowercase a
b	b	98	b	lowercase b
c	c	99	c	lowercase c
d	d	100	d	lowercase d
e	e	101	e	lowercase e
f	f	102	f	lowercase f
g	g	103	g	lowercase g
h	h	104	h	lowercase h
i	i	105	i	lowercase i
j	j	106	j	lowercase j
k	k	107	k	lowercase k
l	l	108	l	lowercase l
m	m	109	m	lowercase m
n	n	110	n	lowercase n
o	o	111	o	lowercase o
p	p	112	p	lowercase p
q	q	113	q	lowercase q
r	r	114	r	lowercase r
s	s	115	s	lowercase s
t	t	116	t	lowercase t
u	u	117	u	lowercase u
v	v	118	v	lowercase v
w	w	119	w	lowercase w
x	x	120	x	lowercase x

# Character Sets    CHAR-9

## DEC Multinational Character Set

Graphic	EDT Symbol	Decimal Value	Abbrev.	Description
y	y	121	y	lowercase y
z	z	122	z	lowercase z
{	{	123	{	opening brace
		124		vertical line
}	}	125	}	closing brace
~	~	126	~	tilde
DEL	<DEL>	127	DEL	delete, rubout
	<X80>	128	---	[reserved]
	<X81>	129	---	[reserved]
	<X82>	130	---	[reserved]
	<X83>	131	---	[reserved]
	<IND>	132	IND	index
	<NEL>	133	NEL	next line
	<SSA>	134	SSA	start of selected area
	<ESA>	135	ESA	end of selected area
	<HTS>	136	HTS	horizontal tab set
	<HTJ>	137	HTJ	horizontal tab set with justification
	<VTS>	138	VTS	vertical tab set
	<PLD>	139	PLD	partial line down
	<PLU>	140	PLU	partial line up
	<RI>	141	RI	reverse index
	<SS2>	142	SS2	single shift 2
	<SS3>	143	SS3	single shift 3
	<DCS>	144	DCS	device control string
	<PU1>	145	PU1	private use 1
	<PU2>	146	PU2	private use 2
	<STS>	147	STS	set transmit state
	<CCH>	148	CCH	cancel character
	<MW>	149	MW	message waiting
	<SPA>	150	SPA	start of protected area
	<EPA>	151	EPA	end of protected area
	<X98>	152	---	[reserved]
	<X99>	153	---	[reserved]
	<X9A>	154	---	[reserved]
	<CSI>	155	CSI	control sequence introducer
	<ST>	156	ST	string terminator
	<OSC>	157	OSC	operating system command
	<PM>	158	PM	privacy message
	<APC>	159	APC	application program command
	<XA0>	160	---	[reserved]

# CHAR-10 Character Sets

## DEC Multinational Character Set

Graphic	EDT Symbol	Decimal Value	Abbrev.	Description
¡	<!!>	161	¡	inverted exclamation mark
¢	<C/>	162	¢	cent sign
£	<L->	163	£	pound sign
	<XA4>	164	---	[reserved]
¥	<Y->	165	¥	yen sign
	<XA6>	166	---	[reserved]
§	<S0>	167	§	section sign
⌘	<X0>	168	⌘	general currency sign
©	<C0>	169	©	copyright sign
<u>a</u>	<a_>	170	<u>a</u>	feminine ordinal indicator
“	<<<>	171	“	angle quotation mark left
	<XAC>	172	---	[reserved]
	<XAD>	173	---	[reserved]
	<XAE>	174	---	[reserved]
	<XAF>	175	---	[reserved]
°	<0^>	176	°	degree sign
±	<+->	177	±	plus/minus sign
2	<2^>	178	2	superscript 2
3	<3^>	179	3	superscript 3
	<XB4>	180	---	[reserved]
μ	</U>	181	μ	micro sign
¶	<P!>	182	¶	paragraph sign, pilcrow
•	<.,^>	183	•	middle dot
	<XB8>	184	---	[reserved]
1	<1^>	185	1	superscript 1
<u>o</u>	<o_>	186	<u>o</u>	masculine ordinal indicator
”	<>>>	187	”	angle quotation mark right
¼	<14>	188	¼	fraction one quarter
½	<12>	189	½	fraction one half
	<XBE>	190	---	[reserved]
¿	<??>	191	¿	inverted question mark
À	<A^>	192	À	uppercase A with grave accent
Á	<A^>	193	Á	uppercase A with acute accent
Â	<A^>	194	Â	uppercase A with circumflex
Ã	<A^>	195	Ã	uppercase A with tilde
Ä	<A" >	196	Ä	uppercase A with umlaut,(diaeresis)
Å	<A*>	197	Å	uppercase A with ring
Æ	<AE>	198	Æ	uppercase AE diphthong
Ç	<C,>	199	Ç	uppercase C with cedilla
È	<E^>	200	È	uppercase E with grave accent
É	<E^>	201	É	uppercase E with acute accent



# Character Sets    CHAR-11

## DEC Multinational Character Set

Graphic	EDT Symbol	Decimal Value	Abbrev.	Description
Ê	<Eˆ>	202	Ê	uppercase E with circumflex
Ë	<E" >	203	Ë	uppercase E with umlaut, (diaeresis)
Ì	<Iˆ>	204	Ì	uppercase I with grave accent
Í	<I' >	205	Í	uppercase I with acute accent
Î	<Iˆ>	206	Î	uppercase I with circumflex
Ï	<I" >	207	Ï	uppercase I with umlaut, (diaeresis)
	<XD0>	208	---	[reserved]
Ñ	<Nˆ>	209	Ñ	uppercase N with tilde
Ò	<Oˆ>	210	Ò	uppercase O with grave accent
Ó	<O' >	211	Ó	uppercase O with acute accent
Ô	<Oˆ>	212	Ô	uppercase O with circumflex
Õ	<Oˆ>	213	Õ	uppercase O with tilde
Ö	<O" >	214	Ö	uppercase O with umlaut, (diaeresis)
Œ	<OE>	215	Œ	uppercase OE ligature
Ø	<O/>	216	Ø	uppercase O with slash
Ù	<Uˆ>	217	Ù	uppercase U with grave accent
Ú	<U' >	218	Ú	uppercase U with acute accent
Û	<Uˆ>	219	Û	uppercase U with circumflex
Ü	<U" >	220	Ü	uppercase U with umlaut, (diaeresis)
Ý	<Y" >	221	Ý	uppercase Y with umlaut, (diaeresis)
	<XDE>	222	---	[reserved]
ß	<ss>	223	ß	German lowercase sharp s
à	<aˆ>	224	à	lowercase a with grave accent
á	<a' >	225	á	lowercase a with acute accent
â	<aˆ>	226	â	lowercase a with circumflex
ã	<a˜>	227	ã	lowercase a with tilde
ä	<a" >	228	ä	lowercase a with umlaut, (diaeresis)
å	<a*>	229	å	lowercase a with ring
æ	<ae>	230	æ	lowercase ae diphthong
ç	<c,>	231	ç	lowercase c with cedilla
è	<eˆ>	232	è	lowercase e with grave accent
é	<e' >	233	é	lowercase e with acute accent
ê	<eˆ>	234	ê	lowercase e with circumflex
ë	<e" >	235	ë	lowercase e with umlaut, (diaeresis)
ì	<iˆ>	236	ì	lowercase i with grave accent
í	<i' >	237	í	lowercase i with acute accent
î	<iˆ>	238	î	lowercase i with circumflex
ï	<i" >	239	ï	lowercase i with umlaut, (diaeresis)
	<XF0>	240	---	[reserved]
ñ	<nˆ>	241	ñ	lowercase n with tilde
ò	<oˆ>	242	ò	lowercase o with grave accent

**CHAR-12    Character Sets**  
**DEC Multinational Character Set**

Graphic	EDT Symbol	Decimal Value	Abbrev.	Description
ó	<o´>	243	ó	lowercase o with acute accent
ô	<oˆ>	244	ô	lowercase o with circumflex
õ	<o˜>	245	õ	lowercase o with tilde
ö	<o¨>	246	ö	lowercase o with umlaut, (diaeresis)
œ	<oe>	247	œ	lowercase oe ligature
ø	<o/>	248	ø	lowercase o with slash
ù	<u`>	249	ù	lowercase u with grave accent
ú	<u´>	250	ú	lowercase u with acute accent
û	<uˆ>	251	û	lowercase u with circumflex
ü	<u¨>	252	ü	lowercase u with umlaut, (diaeresis)
ÿ	<y¨>	253	ÿ	lowercase y with umlaut, (diaeresis)
	<XFE>	254	---	[reserved]
	<XFF>	255	---	[reserved]

ZK-1737/7-84

# Appendix DCL

## DCL Commands

The DCL commands that follow are in alphabetical order. The headline for each command includes the name of the command and the command parameters in their required order. Bracketed parameters are optional.

A comma followed by an ellipsis indicates that you can enter a list containing any number of the preceding item. Most list items must be separated by commas, but some (noted in the documentation) items may be connected by plus signs. Lists of more than one qualifier value must be in parentheses. You can omit the parentheses if the list contains only one value. (The documentation always shows the parentheses.)

Qualifiers are listed in alphabetical order. Qualifiers are global unless otherwise stated. Explanations of positive/negative qualifiers apply to the positive qualifier unless otherwise stated.

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

Requires READ access to the input accounting file.

Collects, records, and reports accounting data.

#### **PARAMETERS**

##### **file-spec**

Specification of the input accounting file. Wildcard characters are allowed. The default is SYS\$MANAGER:ACCOUNTNG.DAT.

#### **QUALIFIERS**

**/ACCOUNT=**(["-"],account-name,...)

**/NOACCOUNT** (default)

Selects only those records matching the specified account names (in the user authorization file) if the first item in the list is not a hyphen enclosed in quotation marks ("-"). If a hyphen enclosed in quotation marks is the first item, excludes those records matching the specified account names.

## **DCL-2    DCL Commands**

### **ACCOUNTING**

**/ADDRESS=([“-”,]node-address,...)**

**/NOADDRESS (default)**

Selects only those records matching the specified remote node addresses if the first item in the list is not a hyphen enclosed in quotation marks (“-”). If a hyphen enclosed in quotation marks is the first item, excludes those records matching the specified node addresses.

**/BEFORE[=time]**

**/NOBEFORE (default)**

Selects only those records created or modified before the specified time. You can specify time as absolute or a combination of absolute and delta times. The default is the current time.

**/BINARY**

**/NOBINARY (default)**

Generates an output file that contains image copies of the input records. By default, the output file contains formatted ASCII records. Incompatible with the /BRIEF, /FULL, and /SUMMARY qualifiers.

**/BRIEF (default)**

**/NOBRIEF**

Generates an output file that contains formatted displays of selected items in the input records as listed below. Incompatible with the /BINARY, /FULL, and /SUMMARY qualifiers.

DATE	Date in the format yyyy mmm dd
DAY	Day of the month (1-31)
HOUR	Hour of the day (0-23)
YEAR	Year
TYPE	Type of record (for example, process or print)
SUBTYPE	Type of process (for example, batch or interactive)

**/ENTRY=([“-”,]queue-entry-number,...)**

**/NOENTRY (default)**

Selects only those records matching the specified queue entry numbers if the first item in the list is not a hyphen enclosed in quotation marks (“-”). If a hyphen enclosed in quotation marks is the first item, excludes those records matching the specified queue entry numbers.

**/FULL**

**/NOFULL (default)**

Generates an output file that contains formatted displays of all the information in the input records. Incompatible with the /BINARY, /BRIEF, and /SUMMARY qualifiers.

**/IDENTIFICATION=([“-”,]process-id,...)**

**/NOIDENTIFICATION (default)**

Selects only those records matching the specified process identifications if the first item in the list is not a hyphen enclosed in quotation marks (“-”). If a hyphen enclosed in quotation marks is the first item, excludes those records matching the specified process identifications.

**/IMAGE=([“-”,]image-name,...)**

**/NOIMAGE (default)**

Selects only those records matching the specified image file names if the first item in the list is not a hyphen enclosed in quotation marks (“-”). If a hyphen enclosed in quotation marks is the first item, excludes those records matching the specified image file names.

**/JOB=([“-”,]job-name,...)**

**/NOJOB (default)**

Selects only those records matching the specified job names if the first item in the list is not a hyphen enclosed in quotation marks (“-”). If a hyphen enclosed in quotation marks is the first item, excludes those records matching the specified job names.

**/LOG**

**/NOLOG (default)**

Displays input file names, selected record counts, and rejected record counts on SYS\$OUTPUT as the command executes.

**/NODE=([“-”,]node-name,...)**

**/NONODE (default)**

Selects only those records matching the specified remote node names if the first item in the list is not a hyphen enclosed in quotation marks (“-”). If a hyphen enclosed in quotation marks is the first item, excludes those records matching the specified remote node names. Do not specify a colon (:) in the node name.

**/OUTPUT[=file-spec]**

**/NOOUTPUT**

Writes the accounting information to the specified file. By default, the output is written to SYS\$OUTPUT. The file name defaults to that of the input file. The file type defaults to LIS for formatted files and DAT for binary files.

**/OWNER=([“-”,]owner-process-id,...)**

**/NOOWNER (default)**

Selects only those records matching the specified owner process identification if the first item in the list is not a hyphen enclosed in quotation marks (“-”). If a hyphen enclosed in quotation marks is the first item, excludes those records matching the specified owner process identifications.

## DCL-4 DCL Commands

### ACCOUNTING

**/PRIORITY=[["-"],priority,...)**

**/NOPRIORITY (default)**

Selects only those records matching the specified priorities if the first item in the list is not a hyphen enclosed in quotation marks ("-"). If a hyphen enclosed in quotation marks is the first item, excludes those records matching the specified priorities.

**/PROCESS=[["-"],process-type,...)**

**/NOPROCESS (default)**

Selects only those records matching the specified process types if the first item in the list is not a hyphen enclosed in quotation marks ("-"). If a hyphen enclosed in quotation marks is the first item, excludes those records matching the specified process types. Possible *process-types* are: BATCH, DETACHED, INTERACTIVE, NETWORK, and SUBPROCESS.

**/QUEUE=[["-"],queue-name,...)**

**/NOQUEUE (default)**

Selects only those records matching the specified queue names if the first item in the list is not a hyphen enclosed in quotation marks ("-"). If a hyphen enclosed in quotation marks is the first item, excludes those records matching the specified queue names.

**/REJECTED[=file-spec]**

**/NOREJECTED (default)**

Writes rejected records (in binary format) to the specified file. The file name defaults to that of the input file. The file type defaults to REJ.

**/REMOTE\_ID=[["-"],remote-id,...)**

**/NOREMOTE\_ID (default)**

Selects only those records matching the specified remote identifications if the first item in the list is not a hyphen enclosed in quotation marks ("-"). If a hyphen enclosed in quotation marks is the first item, excludes those records matching the specified remote identifications.

**/REPORT[=report-item,...]**

**/NOREPORT (default)**

Includes specified items in a summary report; requires the /SUMMARY qualifier. Specify *report-item* as:

BUFFERED_IO	Total buffered I/Os
DIRECT_IO	Total direct I/Os
ELAPSED	Total elapsed time
EXECUTION	Total images executed



FAULTS	Total page faults
GETS	Total RMS GETs
PAGE_FILE	Maximum page file usage
PAGE_READS	Total page read I/Os
PAGES	Total pages printed
PROCESSOR	Total processor time consumed
QIOS	Total QIOs issued
RECORDS	Total records in file (default)
VOLUMES	Total volumes mounted
WORKING_SET	Maximum working set size

**/SINCE[=time]**

**/NOSINCE (default)**

Selects only those accounting records dated after the specified time. You can specify time as absolute or a combination of absolute and delta times. The time defaults to midnight of the current day.

**/SORT=[[-]sort-item,...)**

**/NOSORT (default)**

Sequences the accounting records in the output file. By default, the sequence is the same as that of the input file. You can sequence records according to the ASCII values of any of the following fields in the order specified. The fields are sorted in ascending order unless preceded by a hyphen. Rejected records are not included in the sort. A record that does not contain a specified key field is rejected. Incompatible with /SUMMARY. Specify *sort-item* as:

ACCOUNT	User's account name
ADDRESS	Remote node address
BUFFERED_IO	Buffered I/O count
DIRECT_IO	Direct I/O count
ELAPSED	Elapsed time
ENTRY	Number of batch or print job queue entry
EXECUTION	Image execution count
FAULTS	Page faults
FINISHED	Termination time or time record was finished
GETS	Total RMS GETs
IDENT	Process identification
IMAGE	Image name

## DCL-6 DCL Commands

### ACCOUNTING

JOB	Name of batch or print job
NODE	Remote node name
OWNER	Owner process identification
PAGE_FILE	Peak page file usage
PAGE_READS	Page read I/Os
PAGES	Total pages printed
PRIORITY	Process priority
PROCESS	Process type
PROCESSOR	Processor time consumed
QUEUE	Name of queue
QUEUED	Time batch or print job was queued
QIOS	Total QIOs issued
STARTED	Start time
STATUS	Exit status
TERMINAL	Terminal name
TYPE	Record type
UIC	User identification code
USER	User's name
VOLUMES	Total volumes mounted
WORKING_SET	Peak working set size

**/STATUS=([ "-", ]exit-status,...)**

**/NOSTATUS (default)**

Selects only those records matching the specified exit states if the first item in the list is not a hyphen enclosed in quotation marks ("-"). If a hyphen enclosed in quotation marks is the first item, excludes those records matching the specified exit states.

**/SUMMARY=(summary-item,...)**

**/NOSUMMARY (default)**

Generates an output file that contains formatted displays of selected items in the input records as listed below. Incompatible with the /BINARY, /BRIEF, /FULL, and /SORT qualifiers. Specify *summary-item* as:

ACCOUNT	Account name from the UAF
DATE	Date in the format yyyy mmm dd
DAY	Day of the month (1-31)
HOUR	Hour of the day (0-23)

IMAGE	Image name
JOB	Name of batch job or print job
MONTH	Month of year (1-12)
NODE	Remote node name
PROCESS	Process type
QUEUE	Batch or device queue name
TERMINAL	Terminal name
TYPE	Type of record (for example, logout or batch)

**/TERMINAL=[["-"],terminal-name,...)**

**/NOTERMINAL (default)**

Selects only those records matching the specified terminal names if the first item in the list is not a hyphen enclosed in quotation marks ("-"). If a hyphen enclosed in quotation marks is the first item, excludes those records matching the specified terminal names. Specify the physical device name of the terminal including the colon.

**/TITLE=title**

**/NOTITLE (default)**

Specifies a title to be printed in the center of the first line of a summary report. If the title contains spaces or special characters, enclose it in quotation marks.

**/TYPE=[["-"],record-type,...)**

**/NOTYPE (default)**

Selects only those records matching the specified record types if the first item in the list is not a hyphen enclosed in quotation marks ("-"). If a hyphen enclosed in quotation marks is the first item, excludes those records matching the specified record types. Possible *record-types* are:

FILE	Accounting file forward and backward pointers
IMAGE	Termination of image
LOGFAIL	Unsuccessful conclusion of a login attempt
PRINT	Termination of a print job
PROCESS	Termination of process
SYSINIT	System initialization
UNKNOWN	Any record not recognized as one of the above
USER	Arbitrary user messages

**/UIC=[["-"],uic,...)**

**/NOUIC (default)**

Selects only those records matching the specified UIC if the first item in the list is not a hyphen enclosed in quotation marks ("-"). If a hyphen enclosed in

## DCL-8 DCL Commands

### ACCOUNTING

quotation marks is the first item, excludes those records matching the specified UIC. Wildcard characters can be used in both the group and member fields of the UIC.

**/USER=**(["-"],username,...)

**/NOUSER** (default)

Selects only those records matching the specified user names if the first item in the list is not a hyphen enclosed in quotation marks ("-"). If a hyphen enclosed in quotation marks is the first item, excludes those records matching the specified user names.

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

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

#### PARAMETERS

##### **device-name**

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

##### **logical-name**

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

#### QUALIFIERS

**/GENERIC**

**/NOGENERIC** (default)

Indicates that the first parameter is a device *type* rather than a device *name*. Example device types are: RX50, RD52, TK50, RC25, RCF25, RL02.

**/LOG** (default)

**/NOLOG**

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

## **APPEND input-file-spec,... output-file-spec**

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

### **PARAMETERS**

#### **input-file-spec**

The specification of the input files. Multiple input files are appended to the output file in the order specified; separate file specifications with commas or plus signs. Wildcard characters are allowed.

#### **output-file-spec**

A valid file specification. You must specify at least one field of the file specification. Device and directory default to your current default device and directory. Other fields default to the corresponding field of the first input file specification.

### **QUALIFIERS**

#### **/ALLOCATION=number-of-blocks**

Qualifies output-file-spec.

Forces the initial allocation of the output file to the specified number of 512-byte blocks. If you do not specify the /ALLOCATION qualifier, the initial allocation of the output file is determined by the size of the input file. Relevant only with the /NEW\_VERSION qualifier.

#### **/BACKUP**

#### **/CREATED (default)**

#### **/EXPIRED**

#### **/MODIFIED**

Selects files for the append operation according to the dates of their most recent backups, their creation dates, their expiration dates, or the dates of their last modifications. Relevant only with the /BEFORE and /SINCE qualifiers.

#### **/BEFORE[=time]**

Selects for the append operation only those files dated before the specified time. You can specify time as an absolute time, as a combination of absolute and delta times, or as one of the following keywords: TODAY (default), TOMORROW, or YESTERDAY. Specified with /BACKUP, /CREATED (default), /EXPIRED, or /MODIFIED.

#### **/BY\_OWNER[=uic]**

Selects for the append operation only those files with the specified user identification code. The default user identification code is that of the current process.

## **DCL-10    DCL Commands**

### **APPEND**

#### **/CONFIRM**

#### **/NOCONFIRM (default)**

Issues a request for confirmation before each append operation. The following responses are valid:

YES	Perform the append operation
NO	Do not perform the append operation
TRUE	Perform the append operation
FALSE	Do not perform the append operation
1	Perform the append operation
0	Do not perform the append operation
RETURN	Do not perform the append operation
ALL	Continue execution of the command with no further confirmation prompts
CTRL/Z	Stop execution of the command
QUIT	Stop execution of the command

#### **/CONTIGUOUS**

#### **/NOCONTIGUOUS**

Qualifies output-file-spec.

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

#### **/CREATED**

See /BACKUP.

#### **/EXCLUDE=(file-spec,...)**

Omits the specified files from the append operation. You can include a directory but not a device in the file specifications. Wildcard characters are supported for file specifications. However, you cannot use relative version numbers to exclude a specific version.

#### **/EXPIRED**

See /BACKUP.

#### **/EXTENSION=n**

Qualifies output-file-spec.

Sets the extend quantity default for the output file. Relevant only with the /NEW\_VERSION qualifier.



**/LOG**

**/NOLOG (default)**

Displays file specifications of input and output files as well as the number of blocks or records appended after each append operation. Displays the number of new files created after the entire append operation.

**/MODIFIED**

See /BACKUP.

**/NEW\_VERSION**

**/NONEW\_VERSION (default)**

Qualifies output-file-spec.

Creates a new output file if the specified output file does not exist. If the output file does exist, the /NEW\_VERSION qualifier appends the input file to the output file.

**/PROTECTION={ownership[:access],...}**

Qualifies output-file-spec.

Specifies protection for the output file. Specify ownership as SYSTEM, OWNER, GROUP, or WORLD and access as R, W, E, or D. The default protection is that of the existing output file; if no output file exists, the current default protection applies.

**/READ\_CHECK**

**/NOREAD\_CHECK (default)**

Qualifies input-file-spec.

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

**/SINCE[=time]**

Selects for the append operation only those files dated after the specified time. You can specify time as an absolute time, a combination of absolute and delta times, or as one of the following keywords: TODAY (default), TOMORROW, or YESTERDAY. Specified with /BACKUP, /CREATED (default), /EXPIRED, or /MODIFIED.

**/WRITE\_CHECK**

**/NOWRITE\_CHECK (default)**

Qualifies output-file-spec.

Reads each record in the output file after the record is written to verify that it is written correctly.

## **ASSIGN** equivalence-name,... logical-name

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

### PARAMETERS

#### **equivalence-name**

A character string of 1 to 255 characters. If the string contains other than uppercase alphanumeric, dollar sign, or underscore characters, enclose it in quotation marks (""). Use double quotation marks (""") to denote an actual quotation mark. Specifying more than one equivalence name for a logical name creates a search list.

#### **logical-name**

A character string of 1 to 255 characters. If the string contains other than upper case alphanumeric, dollar sign, or underscore characters, enclose it in quotation marks (""). Use double quotation marks (""") to denote an actual quotation mark. If you terminate *logical-name* with a colon, the system removes the colon. If the logical name is to be entered into the process or system directory logical name tables, then the name may only have from 1 to 31 alphanumeric characters (including the dollar sign and underscore).

### QUALIFIERS

#### **/EXECUTIVE\_MODE**

#### **/SUPERVISOR\_MODE (default)**

#### **/USER\_MODE**

Requires SYSNAM privilege for executive mode.

Specifies the mode of the logical name. If you specify executive mode, but do not have SYSNAM privilege, a supervisor mode logical name is created. The mode of the logical name must be the same as or external to (less privileged than) the mode of the table in which you are placing the name. If you specify a user mode logical name in the process logical name table, that logical name is used for the execution of a single image only; user mode entries are deleted from the logical name table when any image executing in the process exits; that is, after any DCL command or user program that executes an image completes execution.

#### **/GROUP**

#### **/JOB**

#### **/PROCESS (default)**

#### **/SYSTEM**

Require SYSPRV or GRPNAM privilege for group logical names.

Require SYSNAM or SYSPRV privilege for system logical names.

Specifies the table in which the logical name is to be placed. The /GROUP qualifier is synonymous with /TABLE=LNМ\$GROUP. The /JOB qualifier is synonymous with /TABLE=LNМ\$JOB. The /PROCESS qualifier is synonymous

with /TABLE=LNМ\$PROCESS. The /SYSTEM qualifier is synonymous with /TABLE=LNМ\$SYSTEM.

**/LOG (default)**  
**/NOLOG**

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

**/NAME\_ATTRIBUTES[=(keyword,...)]**

Specifies the attributes for a logical name. (By default, no attributes are set. You can specify the following keywords for attributes:

- |          |   |
|----------|---|
| CONFINE  | Does not copy the logical name into a spawned subprocess; Relevant only for logical names in a private table.   |
| NO_ALIAS | Prohibits creation of logical names with the same name in an outer (less privileged) access mode within the specified table; deletes any previously created identical names in an outer access mode within the specified table. |

**/PROCESS**

See /GROUP.

**/SUPERVISOR\_MODE**

See /EXECUTIVE\_MODE.

**/SYSTEM**

See /GROUP.

**/TABLE=name**

Requires WRITE access to the table if the table is shareable.

Specifies the name of the logical name table in which the logical name is to be entered. You can specify user-defined tables (created with the CREATE/NAME\_TABLE command), the process, group, job, or system logical name table, or the process or system logical name directory table. The default is LNМ\$PROCESS. If you specify a table name that translates to more than one table, the logical name is placed in the first table found.

**/TRANSLATION\_ATTRIBUTES[=(keyword,...)]**

Qualifies each equivalence-name.

Specifies attributes of the *equivalence-name*. Possible keywords are:

## **DCL-14    DCL Commands**

### **ASSIGN**

CONCEALED	Indicates that the equivalence string is the name of a concealed device
TERMINAL	Terminates logical name translation after iterative translation of this equivalence string

#### **/USER\_MODE**

See /EXECUTIVE\_MODE.

#### **ASSIGN/MERGE merge-queue-name source-queue-name**

Requires OPER privilege or EXECUTE access to both queues.

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

##### **PARAMETERS**

##### **merge-queue-name**

Name of the queue into which the jobs are being merged.

##### **source-queue-name**

Name of the queue from which the jobs are being removed.

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

Requires OPER privilege or EXECUTE access to both queues.

Assigns, or redirects, a logical queue to a single execution queue.

##### **PARAMETERS**

##### **execution-queue-name[:]**

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

##### **logical-queue-name[:]**

Name of the logical queue.

#### **ATTACH [process-name]**

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

## PARAMETERS

### **process-name**

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

## QUALIFIERS

### **/IDENTIFICATION=pid**

Specifies the process identification (PID) of the process to which terminal control will be transferred. Leading zeros can be omitted. Incompatible with the process-name parameter.

## **BACKUP** input-file-spec,... [output-file-spec,...]

Copies files, saves files on save sets, restores files from save sets, lists files on save sets, and compares files with other files or save sets.

## PARAMETERS

### **input-file-spec**

Specification of a file to be copied or saved, or a save set to be restored. Wildcard characters are allowed in the specifications of regular files, and in those of save-set files on magnetic tape. DECnet node names are allowed only in save-set file specifications.

### **output-file-spec**

Specification of a file to be created by a copy or restore operation, or a save set to be created by a save operation. Wildcard characters are allowed for normal files. No wildcard characters are allowed for save sets. DECnet node names are allowed only in save-set file specifications.

## QUALIFIERS

### **/BACKUP**

### **/CREATED**

### **/EXPIRED**

### **/MODIFIED (default)**

Qualifies input files.

Selects files for the backup operation according to the dates of their most recent backups, their creation dates, their expiration dates, or the dates of their last modifications. Relevant only with the /BEFORE and /SINCE qualifiers.



## **DCL-16    DCL Commands**

### **BACKUP**

#### **/BEFORE=time**

Qualifies input files.

Processes only those files dated earlier than the specified time, which can be an absolute time or one of the following keywords:

BACKUP	The date recorded by a previous BACKUP/RECORD; available only on Files-11 Structure Level 2 volumes
TODAY	The current day at 0 hours
TOMORROW	TODAY plus 24 hours
YESTERDAY	TODAY minus 24 hours

#### **/BLOCK\_SIZE=number-of-bytes**

Qualifies output save sets.

Defines the block size, in bytes, for data records in a BACKUP save set. The acceptable range for n is 2048 through 65,534. If you do not specify this qualifier, the default blocking size for a disk save set is 32,528 bytes; for a magnetic tape save set, 8464 bytes.

#### **/BRIEF (default)**

#### **/FULL**

Lists a brief amount of information on each file saved or restored (specification, size, and creation date), or lists the information in the format of the DIRECTORY /FULL command. Relevant only when the /LIST qualifier is also specified.

#### **/BUFFER\_COUNT=number-of-buffers**

Specifies the number of I/O buffers to be used in the backup operation. The maximum value is 5; the default is 3. Relevant with both disks and magnetic tape.

#### **/COMMENT=string**

Qualifies output save sets.

Places the comment in the save set. The comment is displayed in list operations. If the comment contains spaces or special characters, enclose it in quotation marks ("").

#### **/COMPARE**

Compares two files, two groups of files, or a save set and a file or group of files. In the latter case, the save set must be specified as the first parameter. The default version number is \*, which processes all versions of the file.

#### **/CONFIRM**

Qualifies input files.

For each file being copied or saved, displays a query to which you must respond Y to copy the file — any other response skips the copy or save operation for that file.



**/CRC (default)**

**/NOCRC**

Qualifies input and output save sets.

Performs the Cyclic Redundancy Check on save-set data.

**/CREATED**

See /BACKUP.

**/DELETE**

Deletes input files after they are saved.

**/DENSITY=n**

The /DENSITY qualifier is not applicable to the TK50 tape device.

Specifies the density at which a magnetic tape save set is recorded. Use a value that is supported by the magnetic tape drive. If you do not specify the /DENSITY qualifier, the default density is the current density of the magnetic tape drive.

**/EXCLUDE=(file-spec,...)**

Qualifies input files.

Excludes the specified files from the copy or save operation. Wildcard characters are supported for file specification. However, you cannot use relative version numbers to exclude a specific version. No device names are allowed.

**/EXPIRED**

See /BACKUP.

**/FAST**

Requires WRITE access to [0,0]INDEX.SYS on the volume containing the files being saved, or the volume must be write-locked.

Uses a fast file scan. The /FAST qualifier is ignored if used with the /IMAGE qualifier.

**/FULL**

See /BRIEF.

**/GROUP\_SIZE=number-of-blocks**

Qualifies output save sets.

Defines the number of blocks to be placed in each redundancy group. Using the redundant information, one "uncorrectable" read error in each redundancy group can be corrected. The value must be from 0 through 100; the default is 10; 0 means no redundancy blocks are written.

## **DCL-18    DCL Commands**

### **BACKUP**

#### **/IGNORE=keyword**

Overrides restrictions placed on files. Possible keywords are:

- |           |  |
|-----------|--|
| INTERLOCK | Processes files that otherwise could not be processed because of file access conflicts (specifically, files currently open for writing); requires SYSPRV privilege |
| NOBACKUP  | Saves or copies the contents of files that are marked with the /NOBACKUP qualifier of the SET FILE command   |

#### **/IMAGE**

Requires WRITE access to [0,0]INDEXF.SYS and [0,0]BITMAP.SYS on the volume being processed, or the volume must be write-locked.

Copies, saves, or restores an entire volume. The output volume must be mounted /FOREIGN. The file specification parameters must consist only of the disk names. In save and copy operations, other file specification qualifiers are not allowed.

#### **/INCREMENTAL**

Restores an incremental save set to a disk volume. The output file specification must consist only of the device name. The /INCREMENTAL qualifier assumes /OWNER\_UIC=ORIGINAL, and can be used only on Files-11 Structure Level 2 volumes.

#### **/INITIALIZE (default for /IMAGE)**

#### **/NOINITIALIZE (default for sequential disk volumes)**

Initializes the output volume (in an image copy or a save operation). An image copy operation uses the volume initialization data on the input volume. Any existing data on the output volume is lost.

#### **/INTERCHANGE**

Do not use except where explicitly instructed.

Does not copy access control lists or directories not selected as files. Block size on magnetic tape is limited to 8192 bytes. Magnetic tapes are written using normal error recovery to eliminate bad records on the resulting magnetic tape.

#### **/JOURNAL[=file-spec]**

Maintains a record of save sets, times of creation, and contents. When /JOURNAL is specified during a save operation, information concerning the operation is appended to the specified file. A new file is created if the specified file does not exist. The file specification defaults to SYS\$DISK:BACKUP.BJL. Used with /LIST, the /JOURNAL qualifier displays the contents of the journal. The /BEFORE and /SINCE qualifiers, when used with /JOURNAL /LIST, refer to the time at which the save set (not saved files) was created.

**/LABEL=(label,...)**

Qualifies output-file-specs.

Specifies the volume label for a save set written on magnetic tape or sequential disk. For save sets written on tape, you must use a string of 1 to 6 alphanumeric characters. For save sets written on sequential disk volumes, the string must be 1 to 12 alphanumeric characters.

If you do not specify the */LABEL* qualifier, *label* will be derived from the save-set name. In a multivolume disk save set, the volume set name will be the save-set name. The label of each volume in the volume set will be the label string or the label string derived from the save-set name and followed by a two-digit volume number, starting with 01.

If you specify a list of labels, BACKUP will label save-set volume *n* with label *n* in the list of labels. If the list of labels is shorter than the number of volumes in the save set, BACKUP will generate labels for the remaining volumes using the first label in the list followed by a two-digit relative volume number. Note that for magnetic tape save sets, the volumes are counted, starting (at 01) with the tape on which the current save set started. Magnetic tape volume numbers are not maintained across multiple save sets written on multiple magnetic tapes.

**/LIST[=file-spec]**

Lists information on the contents of a save set. The listing is written to *file-spec*, which defaults to *SYS\$OUTPUT*. This qualifier can stand alone, in which case the first and only parameter must be the name of the save set; or the qualifier can be part of a save or restore operation, in which case the listing reflects the status of the save set after completion of the operation. Do not use */LOG* with */LIST* when the output is directed to *SYS\$OUTPUT*.

**/LOG**

**/NOLOG (default)**

Displays on *SYS\$OUTPUT* the name of each file as it is processed.

**/MODIFIED (default)**

See */BACKUP*.

**/NEW\_VERSION**

Qualifies output-file-specs.

Creates a new version of a file if you attempt to restore or copy a file to a directory that has a file of the same name and version number. Files are processed in decreasing version-number order and created in ascending order, so that version numbers are inverted. Results are unpredictable if */NEW\_VERSION* is used with */COMPARE* or */VERIFY*.

**/OVERLAY**

Qualifies output-file-specs.

Overlays the existing named file rather than allocating new space for the file.

## DCL-20    DCL Commands

### BACKUP

#### **/OWNER\_UIC=[uic]**

Positional qualifier.

Requires SYSPRV privilege or ownership of the UIC to specify with an output file or save set.

As an input file qualifier — Processes only files owned by the specified UIC. A specification of /OWNER\_UIC without a value means your process UIC.

As an output file qualifier — Resets the ownership of the copied or restored file. You can specify a UIC or one of the following keywords:

DEFAULT	Your process UIC
ORIGINAL	Existing UIC of the saved file
PARENT	The UIC of the directory under which the file is being restored

If no /OWNER\_UIC is specified, then it is the same as specifying /OWNER\_UIC=DEFAULT. A /OWNER\_UIC specification without a value means ORIGINAL.

As an output save-set qualifier — Specifies ownership of the save set. The default is your process UIC.

#### **/PHYSICAL**

Ignores any file structure on the volume. Output disks must be the same type as input disks; output disks cannot have a bad block where input disks do not. A save set written using /PHYSICAL can only be read as a physical save set. Output disks must have been mounted /FOREIGN; input disks must have been mounted /FOREIGN or the user must have LOG\_IO privilege. The file specification for a physical volume can contain only a device name. Note that this should only be used on input and output disks proven to have no bad blocks.

#### **/PROTECTION[=(ownership[:access],...)]**

Qualifies output save sets.

Specifies protection for a save set. Specify ownership as SYSTEM, OWNER, GROUP, or WORLD. Specify access as R, W, E, or D. Default protection is that of the current process.

#### **/RECORD**

Requires ownership of the file or SYSPRV privilege.

Records in the file header the backup date for each file processed. Valid only for Files-11 Structure Level 2 volumes.

#### **/REPLACE**

Qualifies output files.

Deletes the existing output file and creates a new file if you attempt to restore or copy a file to a directory that has a file of the same name and version number.

**/REWIND**

**/NOREWIND (default)**

Causes the tape reel to be rewound (/REWIND) or not rewound (/NOREWIND) to beginning-of-tape (BOT) before BACKUP searches for the save-set name specified in the input specifier. Use this qualifier only for magnetic tape save sets.

**/SAVE\_SET**

Qualifies input and output save sets.

Identifies the file specification as a save set, not a normal file. This qualifier is required for save sets on disk.

**/SELECT=(file-spec,...)**

Qualifies input save sets.

Restores only the specified files. Wildcard characters are allowed. No device names are allowed.

**/SINCE=time**

Qualifies input files.

Selects only those files dated after the specified time. You can specify time as an absolute time or as one of the following keywords:

BACKUP	The date recorded by a previous BACKUP/RECORD
TODAY	The current day at 0 hours
TOMORROW	TODAY plus 24 hours
YESTERDAY	TODAY minus 24 hours

**/TRUNCATE**

**/NOTRUNCATE (default)**

Truncates a sequential output file at end-of-file when creating it during a copy or restore operation. By default, a copy or restore operation uses the allocation of the input file to determine the size of the output file.

**/VERIFY**

Verifies data transfers. On file-structured copy operations, /VERIFY compares each file after it has been copied. On physical copy operations, /VERIFY compares the volume after it has been copied. For save or restore operations, /VERIFY compares in a separate pass. Incompatible with the /NEW\_VERSION qualifier.

**/VOLUME=number-of-volume**

Processes a specific disk volume in a disk volume set. Valid only with the /IMAGE qualifier. The /VOLUME qualifier requires only n+1 drives (rather than 2\*n); during a copy or save operation the entire set must be write-locked for consistent results. In save operations the save set contains the segments of



## **DCL-22    DCL Commands**

### **BACKUP**

the files located on the specified volume. The input volume set must be fully mounted. The save set can be restored only with the /VOLUME qualifier. In restore operations the input save set must have been created using the /IMAGE qualifier. In restore operations the output volume is a functionally equivalent copy of the selected relative volume. The input save set can be either an image save set of a full disk volume set or a selected volume save set created with the /VOLUME qualifier. You cannot use the /NOINITIALIZE qualifier in a restore operation with the /VOLUME qualifier. In a selected volume-compare operation between two disk volume sets, both volume sets must be fully mounted.

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

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

#### **PARAMETERS**

##### **label**

Specifies a 1- through 255-alphanumeric character label appearing as the first item on a command line. A label may not contain embedded blanks. When the CALL command is executed, control passes to the command following the specified label. The label can precede or follow the CALL statement in the current command procedure. When you use a label in a command procedure, it must be terminated with a colon. All labels are procedure level dependent except for those labels that define subroutine entry points. These labels are local to the current command procedure file level. Labels for subroutine entry points must be unique.

##### **p1 [p2 [... p8]]**

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

#### **QUALIFIER**

##### **/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 CALL command. No wildcard characters are allowed in the output file specification.



## **CANCEL [process-name]**

Requires ownership of the process, or GROUP or WORLD privilege.

Cancels wakeup requests for a specified process that were scheduled with either the RUN command or the \$SCHDWK system service.

### PARAMETERS

#### **process-name**

Name of the process. The specified process must have the same group number in its UIC as does the current (issuing) process. Defaults to the current process. Ignored if the /IDENTIFICATION qualifier is specified.

### QUALIFIERS

#### **/IDENTIFICATION=pid**

Identifies the process by its process identification (PID). You can omit leading zeros in the PID.

## **CLOSE logical-name[:]**

Closes a file opened with the OPEN command and deassigns the associated logical name. (Files that are opened for reading or writing at DCL command level stay open until explicitly closed with the CLOSE command or until the process terminates.)

### PARAMETERS

#### **logical-name**

Logical name associated with the file.

### QUALIFIERS

#### **/ERROR=label**

Specifies a label in the command procedure to receive control if the CLOSE operation results in an error. Overrides any ON condition action specified and sets \$STATUS to success.

#### **/LOG (default)**

#### **/NOLOG**

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

## **CONNECT virtual-terminal**

Requires that the virtual terminal feature for the system and for your terminal be enabled with the SYSGEN utility.

Connects your physical terminal to a virtual terminal that is connected to another process with your UIC. No other physical terminals may be connected to the virtual terminal.

### **PARAMETERS**

#### **virtual-terminal**

Name of the virtual terminal.

### **QUALIFIERS**

#### **/CONTINUE**

#### **/NOCONTINUE (default)**

Continues execution of your previous process after the connection is established. Incompatible with the /LOGOUT qualifier.

#### **/LOGOUT (default)**

#### **/NOLOGOUT**

Logs out your previous process when the connection is established. Incompatible with the /CONTINUE qualifier.

## **CONTINUE**

Resumes execution of an image or command procedure interrupted by CTRL/Y or CTRL/C. You cannot resume execution of the image if you have entered a command that executes another image or if you have invoked a command procedure.

## **COPY input-file-spec,... output-file-spec**

Creates a new file (or files) from one or more existing files. If device or directory is not specified, your current default device and directory are used.

### **PARAMETERS**

#### **input-file-spec**

Specifications of existing files to be copied. Wildcard characters are allowed. Use a plus sign or a comma to indicate multiple file specifications.

#### **output-file-spec**

Name of the resultant output file. Wildcard characters can be used in the directory name, file name, file type, and/or version number to generate multiple output files. You must specify at least one field in the output file specification.

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

#### QUALIFIERS

##### **/ALLOCATION=number-of-blocks**

Qualifies output-file-spec.

Forces the initial allocation of the output file to the specified number of 512-byte blocks. If you do not specify the /ALLOCATION qualifier, the initial allocation of the output file is determined by the size of the input file.

##### **/BACKUP**

##### **/CREATED (default)**

##### **/EXPIRED**

##### **/MODIFIED**

Selects files for the copy operation according to the dates of their most recent backups, their creation dates, their expiration dates, or the dates of their last modifications. Relevant only with the /BEFORE and /SINCE qualifiers.

##### **/BEFORE[=time]**

Selects only those files with dates that precede the specified time. You can specify time as an absolute time, as a combination of absolute and delta times, or as one of the following keywords: TODAY (default), TOMORROW, or YESTERDAY. Specified with /BACKUP, /CREATED (default), /EXPIRED, or /MODIFIED.

##### **/BY\_OWNER[=uic]**

Selects for the copy operation only those files with the specified user identification code. The default user identification code is that of the current process.

##### **/CONCATENATE (default)**

##### **/NOCONCATENATE**

Creates one output file from multiple input files when wildcard characters are not used in the output file specification. A specification of /NOCONCATENATE generates multiple output files. Files from Files-11 Structure Level 2 disks are concatenated in alphanumeric order; if you specify a wildcard in the file version field, files are copied in descending order by version number. Files from Files-11 Structure Level 1 disks are concatenated in random order.

##### **/CONFIRM**

##### **/NOCONFIRM (default)**

Issues a request for confirmation before each copy operation. The following responses are valid:

## **DCL-26    DCL Commands**

### **COPY**

YES	Perform the copy operation
NO	Do not perform the copy operation
TRUE	Perform the copy operation
FALSE	Do not perform the copy operation
1	Perform the copy operation
0	Do not perform the copy operation
RETURN	Do not perform the copy operation
ALL	Continue execution of the command with no further confirmation prompts
CTRL/Z	Stop execution of the command
QUIT	Stop execution of the command

#### **/CONTIGUOUS**

#### **/NOCONTIGUOUS**

Qualifies output-file-spec.

Specifies that the output file must occupy contiguous physical disk blocks. By default, the COPY command creates an output file in the same format as the corresponding input file and does not report an error if not enough space exists for a contiguous allocation.

The /CONTIGUOUS qualifier has no effect when you copy files to or from tapes because the size of the file on tape cannot be determined until after it is copied to the disk. If you copy a file from a tape and want the file to be contiguous, use the COPY command twice: once to copy the file from the tape, and a second to create a contiguous file.

#### **/CREATED**

See /BACKUP.

#### **/EXCLUDE=(file-spec,...)**

Omits the specified files from the copy operation. You can include a directory but not a device in the file specifications. Wildcard characters are supported for file specifications. However, you cannot use relative version numbers to exclude a specific version.

#### **/EXPIRED**

See /BACKUP.

#### **/EXTENSION=n**

Qualifies output-file-spec.

Sets the extend quantity default for the output file.

**/LOG**

**/NOLOG (default)**

Displays file specifications of input and output files as well as the number of blocks or records copied after each copy operation. Displays the number of files created after the entire copy operation.

**/MODIFIED**

See /BACKUP.

**/OVERLAY**

**/NOOVERLAY (default)**

Qualifies output-file-spec.

Overlays the existing specified file rather than allocating new space for the file.

**/PROTECTION=(ownership[:access],...)**

Qualifies output-file-spec.

Specifies protection for the output file. Specify ownership as SYSTEM, OWNER, GROUP, or WORLD and access as R, W, E, or D. The default protection is that of the existing output file; if no output file exists, the current default protection applies.

**/READ\_CHECK**

**/NOREAD\_CHECK (default)**

Qualifies input-file-spec.

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

**/REPLACE**

**/NOREPLACE (default)**

Qualifies output-file-spec.

Deletes an existing file and creates a new file (allocating new space) if a version number is specified in the output file. If no version number is specified in the output file specification, a new version of the output file is created (regardless of /REPLACE). If an existing version number is specified in the output file specification without /REPLACE, an error occurs.

**/SINCE[=time]**

Selects only those files dated after the specified time. You can specify time as an absolute time, a combination of absolute and delta times, or as one of the following keywords: TODAY (default), TOMORROW, or YESTERDAY. Specified with /BACKUP, /CREATED (default), /EXPIRED, or /MODIFIED.

**/TRUNCATE**

**/NOTRUNCATE (default)**

Qualifies output-file-spec.

Truncates the output file at end-of-file during the copy operation. By default, the size of the output file is determined by the allocation of the input file.

**/VOLUME=relative-volume-number**

Qualifies output-file-spec.

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

**/WRITE\_CHECK**

**/NOWRITE\_CHECK (default)**

Qualifies output-file-spec.

Reads each record in the output files after the record is written to verify that it is written correctly.

**CREATE file-spec,...**

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

**PARAMETERS**

**file-spec**

Specification of the file being created. Wildcard characters are not allowed. The file name and the file type default to null strings. If the specified file already exists, a new version is created.

**QUALIFIERS**

**/LOG**

**/NOLOG (default)**

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

**/OWNER\_UIC=uic**

Requires SYSPRV privilege for a UIC other than your own.

Specifies an owner UIC for the file.

**/PROTECTION=(ownership[:access],...)**

Specifies protection for the file. Specify ownership as SYSTEM, OWNER, GROUP, or WORLD and protection as R, W, E, or D. The default protection is the current default protection.



**/VOLUME=relative-volume-number**

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

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

Requires WRITE access to the master file directory (MFD) to create a first-level directory.

Creates one or more new directories or subdirectories.

**PARAMETERS**

**directory-spec**

Valid directory specification optionally preceded by a device name (and colon); the directory defaults to the current default. Wildcard characters are not allowed.

**QUALIFIERS**

**/LOG**

**/NOLOG (default)**

Displays the directory specification of each directory created as the command executes.

**/OWNER\_UIC=[uic]**

Requires SYSPRV privilege for a UIC other than your own.

Specifies an owner UIC for the directory. The default is your UIC. You can specify the keyword PARENT in place of a UIC to mean the UIC of the parent directory. If a user with privileges creates a subdirectory, the default is that the owner of the subdirectory will be the owner of the parent directory (or the owner of the Master File Directory if creating a main level directory). If you do not specify the /OWNER\_UIC qualifier when creating a directory, the command assigns ownership as follows: (1) if you specify the directory name in either alphanumeric or subdirectory format, the default is your UIC (unless you are privileged in which case the UIC defaults to the parent directory); (2) if you specify the directory in UIC format, the default is the specified UIC.

**/PROTECTION=(ownership[:access],...)**

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

**/VERSION\_LIMIT=limit**

Specifies the number of versions of any one file that can exist in the directory. If you go over the limit, the system deletes the lowest numbered version. A specification of 0 means no limit. A maximum limit of approximately 60

## **DCL-30     DCL Commands**

### **CREATE/DIRECTORY**

versions applies no matter the specification. The default is the limit for the parent directory.

#### **/VOLUME=relative-volume-number**

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

### **CREATE/NAME\_TABLE table**

Creates a logical name table.

#### **PARAMETERS**

##### **table**

A string of from 1 to 31 characters that identifies the logical name table. The string can include alphanumeric characters, the dollar sign, and the underscore.

#### **QUALIFIERS**

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

Specifies attributes for the table. By default, no attributes are set. Possible keywords are:

- |           |   |
|-----------|---|
| CONFINE   | Does not copy the table name or the logical names contained in the table into a spawned subprocess; relevant only for process-private tables.   |
| NO_ALIAS  | Prohibits creation of table names with the same name in an outer (less privileged) access mode in the same logical name table directory; deletes any previously created identical table names in an outer access mode in the same logical name table directory. |
| SUPERSEDE | If a name exists within the appropriate directory at the indicated access mode, it is deleted and a new table is created.   |

##### **/EXECUTIVE\_MODE**

##### **/SUPERVISOR\_MODE (default)**

##### **/USER\_MODE**

Requires SYSNAM privilege for executive mode.

Specifies the access mode of the table. If you specify executive mode without having SYSNAM privilege, a supervisor mode logical name table is created.

##### **/LOG (default)**

##### **/NOLOG**

Specifies whether or not an informational message is generated when the SUPERSEDE attribute is specified or when the table already exists but the SUPERSEDE attribute is not specified.

**/PARENT\_TABLE=table**

Requires SYSPRV and ENABLE access to specify a shareable table.

Specifies the name of the parent table, which must have the same access mode or an access mode inner to the access mode of the table you are creating. Defaults to LNM\$PROCESS\_DIRECTORY. The parent table determines whether the table is private or shareable and where the quota of the new table comes from.

**/PROTECTION=(ownership[:access],...)**

Applies the specified protection to shareable name tables. The ownership categories are SYSTEM, OWNER, GROUP, WORLD; the access categories are R (read), W (write), E (enable), and D (delete). The default protection is (SYSTEM:RWED,OWNER:RWED,GROUP:,WORLD:)

**/QUOTA=number-of-bytes**

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

**/SUPERVISOR\_MODE (default)**

See /EXECUTIVE\_MODE.

**/USER\_MODE**

See /EXECUTIVE\_MODE.

**DEALLOCATE [device-name]**

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

**PARAMETERS**

**device-name**

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

## DCL-32    DCL Commands

### DEALLOCATE

#### QUALIFIERS

##### **/ALL**

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

#### **DEASSIGN [logical-name[:]]**

Deletes a logical name. Logical names in private tables are deleted automatically when your process terminates. All logical names in the job table and the job table itself are deleted when your process terminates. User mode logical names in the process table are deleted automatically when the next image exits. All other logical names in shareable tables remain unless explicitly deassigned. All names in descendant tables are deleted when the parent table logical name is deassigned.

#### PARAMETERS

##### **logical-name**

The logical name to be deleted. Names containing other than alphanumeric, dollar sign, or underscore characters must be enclosed in quotation marks (""). If the name ends with a colon, you must specify two colons. Incompatible with the /ALL qualifier.

#### QUALIFIERS

##### **/ALL**

Deletes all logical names in the same or an outer (less privileged) access mode. Incompatible with the logical-name parameter.

##### **/EXECUTIVE\_MODE**

##### **/SUPERVISOR\_MODE (default)**

##### **/USER\_MODE**

Requires SYSNAM privilege to deassign executive mode.

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

##### **/GROUP**

##### **/JOB**

##### **/PROCESS (default)**

##### **/SYSTEM**

Group logical names require GRPNAM or SYSPRV privilege.

System logical names require SYSNAM or SYSPRV privilege.

Specifies the table in which the logical name resides. The /GROUP qualifier is synonymous with /TABLE=LNМ\$GROUP. The /JOB qualifier is synonymous with /TABLE=LNМ\$JOB. The /PROCESS qualifier is synonymous with

**/TABLE=LNМ\$PROCESS.** The **/SYSTEM** qualifier is synonymous with **/TABLE=LNМ\$SYSTEM.**

**/PROCESS**

See **/GROUP.**

**/SUPERVISOR\_MODE**

See **/EXECUTIVE\_MODE.**

**/SYSTEM**

See **/GROUP.**

**/TABLE=name**

Requires **WRITE** access to the table to delete a shareable logical name.

Requires **SYSPRV** or **DELETE** access to delete a shareable logical name table.

Specifies the table from which the logical name is to be deleted. Defaults to **LNМ\$PROCESS.** The table can be the process, group, job, or system table, one of the directory tables, or the name of a user-created table.

**/USER\_MODE**

See **/EXECUTIVE\_MODE.**

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

Requires the **OPER** privilege or **EXECUTE** access to the queue.

Deassigns a logical queue from a physical device queue. Any jobs in the queue are left pending until the logical queue is reassigned to another device queue.

**PARAMETERS**

**logical-queue-name[:]**

Name of the logical queue.

**DECK**

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



## **DCL-34    DCL Commands**

### **DECK**

#### **QUALIFIERS**

##### **/DOLLARS[=string]**

Sets the end-of-file indicator to the specified string of 1 through 15 characters. Enclose the string in quotation marks if it contains literal lowercase letters, multiple blanks, or tabs. You must specify the EOD command to signal the end of the stream if you do not specify the /DOLLARS qualifier.

#### **DEFINE logical-name equivalence-name,...**

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

#### **PARAMETERS**

##### **equivalence-name**

A character string of 1 to 255 characters. If the string contains other than uppercase alphanumeric, dollar sign, or underscore characters, enclose it in quotation marks ("). Use double quotation marks (") to denote an actual quotation mark. Specifying more than one equivalence name for a logical name creates a search list.

##### **logical-name**

A character string of 1 through 255 characters. If the string contains other than upper case alphanumeric, dollar sign, or underscore characters, enclose it in quotation marks ("). Use double quotation marks (") to denote an actual quotation mark. If the logical name is to be entered into the process or system directory logical name tables, then the name may only have from 1 to 31 alphanumeric characters (including the dollar sign and underscore).

#### **QUALIFIERS**

##### **/EXECUTIVE\_MODE**

##### **/SUPERVISOR\_MODE (default)**

##### **/USER\_MODE**

Executive mode requires SYSNAM privilege.

Specifies the mode of the logical name. If you specify executive mode without SYSNAM privilege, a supervisor mode logical name is created. The mode of the logical name must be the same as or external to (less privileged than) the mode of the table in which you are placing the name.



**/GROUP**

**/JOB**

**/PROCESS (default)**

**/SYSTEM**

Group logical names require GRPNAM or SYSPRV privilege.

System logical names require SYSNAM or SYSPRV privilege.

Specifies the table in which the logical name is to be placed. The /GROUP qualifier is synonymous with /TABLE=LNМ\$GROUP. The /JOB qualifier is synonymous with /TABLE=LNМ\$JOB. The /PROCESS qualifier is synonymous with /TABLE=LNМ\$PROCESS. The /SYSTEM qualifier is synonymous with /TABLE=LNМ\$SYSTEM.

**/LOG (default)**

**/NOLOG**

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

**/NAME\_ATTRIBUTES[=(keyword,...)]**

Specifies the attributes for a logical name. (By default, no attributes are set.)

Possible keywords are:

**CONFINE** Does not copy the logical name into a spawned subprocess; relevant only for logical names in a private table.

**NO\_ALIAS** Prohibits creation of logical names with the same name in a less privileged access mode within the specified table; deletes any previously created identical names in an outer access mode within the specified table.

**/PROCESS**

See /GROUP.

**/SUPERVISOR\_MODE**

See /EXECUTIVE\_MODE.

**/SYSTEM**

See /GROUP.

**/TABLE=name**

Requires WRITE access to the table if the table is shareable.

Specifies the name of the logical name table in which the logical name is to be entered. You can specify user-defined tables (created with the CREATE/NAME\_TABLE command), the process, job, group, or system logical name table, or the process or system logical name directory tables. The default is LNМ\$PROCESS. If you specify a table name that translates to more than one table, the logical name is placed with the first table found.

## **DCL-36    DCL Commands**

### **DEFINE**

#### **/TRANSLATION\_ATTRIBUTES[=keyword,...]**

Qualifies each equivalence string.

Specifies attributes of the equivalence strings. Possible keywords are:

CONCEALED	Indicates that the equivalence string is the name of a concealed device
TERMINAL	Terminates logical name translation after translation of this equivalence string

#### **/USER\_MODE**

See /EXECUTIVE\_MODE.

### **DEFINE/CHARACTERISTIC name number**

Requires OPER privilege.

Assigns a numeric value to a queue characteristic. The characteristic is created if it does not exist. Used in conjunction with the /CHARACTERISTIC qualifier of the PRINT command.

#### **PARAMETERS**

##### **name**

The name of an existing characteristic or a string of 1 through 31 characters that defines a new characteristic. The character string can include any uppercase and lowercase letters, digits, the dollar sign and the underscore, and must include at least one alphabetic character.

##### **number**

An integer in the range 0 through 127.

### **DEFINE/FORM form-name number**

Requires OPER privilege.

Assigns a numeric value to a print forms name. Used in conjunction with the /FORM qualifier of the PRINT command.

#### **PARAMETERS**

##### **form-name**

The name of an existing forms type or a string of from 1 to 31 characters to define a new forms type. The character string can include any uppercase and lowercase letters, digits, the dollar sign and the underscore, and must include at least one alphabetic character.

##### **number**

An integer in the range 0 through 999.

## QUALIFIERS

### **/DESCRIPTION=string**

Specifies a string of from 1 to 255 characters to further describe the form. If the string contains alphanumeric, underscore, or dollar sign characters, it must be enclosed in quotation marks ("). The default string is the name specified in the DEFINE/FORM command.

### **/LENGTH=page-length**

Specifies the physical length of a forms page as an integer in the range 1 through 255. Defaults to 66. The print symbiont sets the page length of the device equal to the form length. This enables the driver to compute the number of line feeds for devices lacking mechanical form feed.

### **/MARGIN=(keyword,...)**

Specifies one or more margin options. Possible keywords are:

- |                  |  |
|------------------|--|
| BOTTOM= <i>n</i> | Specifies the number of blank lines between the end of the print image area and the end of the physical page; the value of <i>n</i> must be between 0 and the value of the /LENGTH parameter and defaults to 6 |
| LEFT= <i>n</i>   | Specifies the number of columns between the leftmost printing position and the print image area; the value of <i>n</i> must be between 0 and the value of the /WIDTH parameter and defaults to 0               |
| RIGHT= <i>n</i>  | Specifies the number of columns between the /WIDTH parameter and the image area; the value of <i>n</i> must be between 0 and the value of the /WIDTH parameter and defaults to 0                               |
| TOP= <i>n</i>    | Specifies the number of blank lines between the top of the physical page and the top of the print image; the value of <i>n</i> must be between 0 and the value of the /LENGTH parameter and defaults to 0      |

### **/PAGE\_SETUP=(module,...)**

#### **/NOPAGE\_SETUP (default)**

Specifies one or more modules that set up the device before every page. The modules are located in the device control library. When a new page is detected, the system extracts the appropriate modules from the device control library and copies them to the printer before the page is printed.

### **/SETUP=(module,...)**

#### **/NOSETUP (default)**

Specifies one or more modules in the device control library that set up the device appropriately for the specified form. When the form is mounted, the system extracts the specified module from the device control library and copies it to the printer before the file is printed.

## **DCL-38    DCL Commands**

### **DEFINE/FORM**

#### **/SHEET\_FEED**

#### **/NOSHEET\_FEED (default)**

Specifies that print jobs pause at the end of every physical page so that a new piece of paper can be inserted.

#### **/STOCK=type**

Specifies the type of paper stock to be associated with the form as a string of from 1 through 31 characters. The string must consist of alphanumeric, underscore, and dollar sign characters.

#### **/TRUNCATE (default)**

#### **/NOTRUNCATE**

Discards any characters that exceed the current line length (specified by /WIDTH and /MARGIN=RIGHT). Incompatible with the /WRAP qualifier. If you specify both /NOTRUNCATE and /NOWRAP, the printer prints as many characters on a line as possible.

#### **/WIDTH=width**

Specifies the physical width of the paper in terms of columns or character positions as an integer in the range 0 through 65535. Defaults to 132. The /MARGIN=RIGHT qualifier overrides the /WIDTH qualifier.

#### **/WRAP**

#### **/NOWRAP (default)**

Causes lines that exceed the current line length (specified by /WIDTH and /MARGIN=RIGHT) to wrap onto the next line. Incompatible with the /TRUNCATE qualifier. If you specify both /NOWRAP and /NOTRUNCATE, the printer prints as many characters on a line as possible.

## **DEFINE/KEY key-name equivalence-string**

Equates a key on the terminal keyboard to a character string.

## PARAMETERS

### key-name

The name of the key to be defined. Permissible keys are as follows:

Key Name	VT100 Key	VT200 Key
PF1	PF1	PF1
PF2	PF2	PF2
PF3	PF3	PF3
PF4	PF4	PF4
KP0, KP1-KP9	Keypad 0-9	Keypad 0-9
PERIOD	Period Key	Period Key
COMMA	Comma Key	Comma Key
MINUS	Minus Key	Minus Key
ENTER	Enter Key	Enter Key
FIND, INSERT HERE	—	Find, Insert Here
REMOVE, SELECT	—	Remove, Select
PREV_SCREEN	—	Prev Screen
NEXT_SCREEN	—	Next Screen
HELP, DO	—	Help(F15), Do(F16)
F6-F20	—	Function Keys F6-F20

**NOTE:** You cannot define the UP and DOWN arrow keys or function keys F1 through F5. You must issue the SET TERMINAL/NOLINE\_EDITING command before defining the LEFT and RIGHT arrow keys and function keys F6 through F14. You can also press CTRL/V to enable keys F7 through F14, but CTRL/V will not enable the F6 key.

### equivalence-string

A character string to replace the key as input. Enclose the string in quotation marks to preserve spaces and lowercase characters.

## QUALIFIERS

### /ECHO (default)

### /NOECHO

Displays the equivalence string on the screen after the key has been pressed. The /NOECHO qualifier is incompatible with the /NOTERMINATE qualifier.

### /ERASE

### /NOERASE (default)

Erases the current line before the key translation is inserted.

## **DCL-40    DCL Commands**

### **DEFINE/KEY**

**/IF\_STATE=(state,...)**

**/NOIF\_STATE (default)**

Specifies a list of one or more states, one of which must be in effect for the key definition to work. The /NOIF\_STATE has the same meaning as /IF\_STATE=current\_state.

**/LOCK\_STATE**

**/NOLOCK\_STATE (default)**

Specifies that the state set by the /SET\_STATE qualifier remain in effect until explicitly changed. (By default, the /SET\_STATE qualifier is in effect only for the next definable key you press or the next read-terminating character that you type.) Relevant only with the /SET\_STATE qualifier.

**/LOG (default)**

**/NOLOG**

Displays a message indicating that the key definition has been successfully created.

**/SET\_STATE=state**

**/NOSET\_STATE (default)**

Specifies a new state to be set when the specified key is pressed. (By default, the current locked state is reset when the key is pressed.) Specify the state as a character string enclosed in quotation marks (").

**/TERMINATE**

**/NOTERMINATE (default)**

Specifies that the key definition be processed immediately when the key is pressed (equivalent to typing the string and pressing RETURN). By default, you can press other keys before the definition is processed.

## **DELETE file-spec,...**

Requires DELETE access to the file or sufficient privilege to override the protection.

Deletes a file or files.

### **PARAMETERS**

**file-spec,...**

Specifications of files being deleted. Wildcard characters can be used. The plus sign can be used in place of the comma between file specifications. Version numbers must be specified on all file specifications. A null version number (a semicolon without the number) or a version number of zero deletes the latest version. A wildcard version number deletes all versions. File specifications after the first one can omit all parts except the version number, and the defaults will come from the preceding specification. The first file specification can omit



only the device and directory; the defaults will come from your current device /directory. Files cannot be deleted from tape devices.

QUALIFIERS

**/BACKUP**

**/CREATED (default)**

**/EXPIRED**

**/MODIFIED**

Selects files for the delete operation according to the dates of their most recent backups, their creation dates, their expiration dates, or the dates of their last modifications. Relevant only with the /BEFORE and /SINCE qualifiers.

**/BEFORE[=time]**

Deletes only those files with dates that precede the specified time. You can specify time as an absolute time, a combination of absolute and delta times, or as one of the following keywords: TODAY (default), TOMORROW, or YESTERDAY. Specified with /BACKUP, /CREATED (default), /EXPIRED, or /MODIFIED.

**/BY\_OWNER[=uic]**

Deletes only those files with the specified user identification code. The default for *uic* is that of the current process.)

**/CONFIRM**

**/NOCONFIRM (default)**

Issues a request for confirmation before each deletion. The following responses are valid:

YES	Delete the file
NO	Do not delete the file
TRUE	Delete the file
FALSE	Do not delete the file
1	Delete the file
0	Do not delete the file
RETURN	Do not delete the file
ALL	Continue execution of the command with no further confirmation prompts
CTRL/Z	Stop execution of the command
QUIT	Stop execution of the command

**/CREATED**

See /BACKUP.

## **DCL-42    DCL Commands**

### **DELETE**

#### **/ERASE**

##### **/NOERASE (default)**

Erases the specified files from the disk so that the deleted data does not exist physically. By default, a file is just marked as deleted.

##### **/EXCLUDE(=file-spec,...)**

Excludes the specified files from deletion. You can include a directory name but not a device name in the file specifications. Wildcard characters are supported for file specifications. However, you cannot use relative version numbers to exclude a specific version.

#### **/EXPIRED**

See /BACKUP.

#### **/LOG**

##### **/NOLOG (default)**

Displays the file specification of each file deleted as the command executes.

#### **/MODIFIED**

See /BACKUP.

##### **/SINCE[=time]**

Selects for deletion only those files dated after the specified time. You can specify time as an absolute time, a combination of absolute and delta times, or one of the following keywords: TODAY (default), TOMORROW, or YESTERDAY. Specified with /BACKUP, /CREATED (default), /EXPIRED, or /MODIFIED.

## **DELETE/CHARACTERISTIC name**

Requires OPER privilege.

Deletes a queue characteristic.

### **PARAMETERS**

#### **name**

The name of the characteristic.

## **DELETE/ENTRY =(job-number,...) queue-name[:]**

Requires OPER privilege, EXECUTE access to the queue, or DELETE access to the job.

Deletes one or more print or batch jobs from a queue. The jobs can be in progress or waiting in the queue. *job-number* specifies the job number of the job being deleted.

PARAMETERS

**queue-name[:]**

Name of the queue handling the job to be deleted.

**DELETE/FORM form-name**

Requires OPER privilege.

Deletes a form type for a printer or a terminal queue. When you delete a form definition, you must ensure that no outstanding references to the form exist in queues that have been mounted with the form or by jobs requesting that form.

PARAMETERS

**form-name**

The name of the form.

**DELETE/INTRUSION\_RECORD source**

Requires CMKRNL and SECURITY privileges.

Removes an entry from the breakin database.

PARAMETERS

**source**

Source field of the entry being removed from the breakin database.

**DELETE/KEY [key-name]**

Deletes key definitions.

PARAMETERS

**key-name**

The name of the key to be deleted. Incompatible with the /ALL qualifier.

QUALIFIERS

**/ALL**

Deletes all key definitions in the specified (or, by default, the current) state. Incompatible with the key-name parameter.

**/LOG (default)**

**/NOLOG**

Displays a message indicating the deletion has taken place.

**DCL-44     DCL Commands**  
**DELETE/KEY**

**/STATE=(state,...)**

**/NOSTATE (default)**

Specifies the name of the state for which the specified key definition is to be deleted. The default state is the current state.

**DELETE/QUEUE queue-name[:]**

Requires OPER privilege.

Deletes a print or batch queue and all the jobs in the queue. The specified queue must have been stopped.

**PARAMETERS**

**queue-name[:]**

Name of the queue being deleted.

**DELETE/SYMBOL [symbol-name]**

Deletes a symbol or symbols.

**PARAMETERS**

**symbol-name**

Name of the symbol being deleted. A name is required unless the /ALL qualifier is specified. Incompatible with the /ALL qualifier.

**QUALIFIERS**

**/ALL**

Deletes all symbols at the specified level. The /ALL qualifier is incompatible with the *symbol-name* parameter.

**/GLOBAL**

**/LOCAL (default)**

Specifies the level of the symbol.

**/LOG**

**/NOLOG (default)**

Displays the symbols being deleted.

**DEPOSIT** location=data,...

Requires user mode READ and WRITE access to the virtual memory location.

Replaces the contents of the specified locations in virtual memory and displays the new contents. If the specified address can be read but not written by the current access mode, the original contents are displayed; if the specified address can be neither read nor written, asterisks are displayed in the data field. The DEPOSIT command maintains a pointer at that location (at the byte following the last byte modified).

**PARAMETERS**

**location**

A virtual address or a range of virtual addresses (where the second address is larger than the first). A location can be any valid integer expression containing an integer value, a symbol name, a lexical function, or a combination of these entities. Radix qualifiers determine the radix in which the address is interpreted; hexadecimal is the initial default radix. Symbol names are always interpreted in the radix in which they were defined. The radix operators %X, %D, or %O can precede the location. A hexadecimal value must begin with a number (or be preceded by %X).

**data**

The data to be deposited into the specified locations; the data is initially interpreted as hexadecimal and deposited in longwords by default.

**QUALIFIERS**

**/ASCII**

Specifies that the data is ASCII. Only one data item is allowed; all characters to the right of the equal sign are considered to be part of a single string. Unless they are enclosed within quotation marks, characters are converted to uppercase; all blanks are compressed into one blank.

**/BYTE**

**/LONGWORD**

**/WORD**

Deposits the data in bytes, longwords, or words. The initial default is longwords.

**/DECIMAL**

**/HEXADECIMAL**

**/OCTAL**

Interprets the data as decimal, hexadecimal, or octal. The initial default is hexadecimal.

**/HEXADECIMAL**

See /DECIMAL.

**DCL-46     DCL Commands**  
**DEPOSIT**

**/LONGWORD**

See /BYTE.

**/OCTAL**

See /DECIMAL.

**/WORD**

See /BYTE.

**DIFFERENCES** input1-file-spec [input2-file-spec]

Compares two files, displaying the records that do not match.

**PARAMETERS**

**input1-file-spec**

Specification of the first file being compared. Wildcard characters are not allowed.

**input2-file-spec**

Specification of the second file being compared. Unspecified fields default to the corresponding fields in input1-file-spec. If the two file specifications are the same or input2-file-spec is omitted, the version number defaults to the next lower version of input1-file-spec.

**QUALIFIERS**

**/CHANGE\_BAR=[(change-char)[,(NO)NUMBER]]**

Marks with the specified character each line in the input1 file that differs from the corresponding line in the input2 file. The change bar character defaults to an exclamation point (!) for ASCII output. If you specify hexadecimal or octal output (see /MODE qualifier), the change bar character is ignored and differences are marked by a "\*\*\*\*CHANGE\*\*\*\*" string. The keyword NONUMBER suppresses line numbers in the listing. If only one option is specified, the parentheses can be omitted.

**/COMMENT\_DELIMITER[(character,...)]**

Ignores lines starting with a specified comment character. If the comment character is an exclamation point or semicolon, it can appear anywhere in the line and characters to the right of the character are ignored. If you specify just one character, you can omit the parentheses. Lowercase characters are automatically converted to uppercase unless they are enclosed in quotation marks. Special DCL characters (such as ! and ,) must be enclosed in quotation marks. You can specify up to 32 comment characters by typing the character itself or one of the following keywords (keywords can be abbreviated provided that the resultant keyword is not ambiguous and has at least two characters).



COLON	Colon (:)
COMMA	Comma (,)
EXCLAMATION	Exclamation point (!)
FORM_FEED	Form feed
LEFT	Left bracket ([)
RIGHT	Right bracket (])
SEMI_COLON	Semicolon (;)
SLASH	Slash (/)
SPACE	Space
TAB	Tab

The following characters are the default comment delimiters for files with the specified file types.

B2S, B32, BAS, BLI	!
CBL, CMD	! and ;
COB	* or / in the first column
COM, COR	!
FOR	! anywhere and C, D, c, d in the first column
HLP	!
MAC, MAR	;
R32, REQ	!

**/IGNORE=(keyword,...)**

Inhibits the comparison of the specified characters, strings, or records, and formats the output file. If you specify only one keyword, you can omit the parentheses. The first set of keywords determines what, if anything, is ignored during file comparison; the second set of keywords determines whether or not ignored characters are included in the output.

BLANK_LINES	Blank lines between data lines.
COMMENTS	Data following a comment character.
FORM_FEEDS	Form feed character.
HEADER[=n]	First <i>n</i> records of the file, beginning with a record whose first character is a form feed. The first record is not ignored if the only character it contains is a form feed. (N indicates the number of records and defaults to 2. A record with a single form feed is not counted.)
TRAILING_SPACES	Space and tab characters at the end of a data line
SPACING	Extra blank spaces or tabs within data lines.

## DCL-48 DCL Commands

### DIFFERENCES

EDITED                      Omits ignored characters from the output records.  
EXACT                       Includes ignored characters in the output records.  
PRETTY                      Formats output records.

If you specify /PARALLEL, output records are always formatted. To format output records, specify

Character	Formatted Output
Tab (CTRL/I)	1-8 spaces
RETURN (CTRL/M)	<CR>
Line feed (CTRL/J)	<LF>
Vertical tab (CTRL/K)	<VT>
Form feed (CTRL/L)	<FF>
Other nonprinting characters	. (period)

#### /MATCH=*n*

Specifies the number of records (*n*) that should indicate matching data after a difference is found; defaults to 3.

#### /MAXIMUM\_DIFFERENCES=*n*

Terminates DIFFERENCES after a specified number of differences (*n*) is found.

#### /MERGED[=*n*]

Specifies that the output file contain a merged list of differences with the specified number (*n*) of matched records listed after each group of unmatched records. The specified number (*n*) must be less than or equal to the number specified in the /MATCH qualifier; defaults to 1. If neither /MERGED nor /SEPARATED nor /PARALLEL is specified, the resulting output is merged, with one matched record following each unmatched record.

#### /MODE=(*radix*,...)

Specifies the format of the output as follows (keywords may be abbreviated): ASCII (default), HEXADECIMAL, or OCTAL.

If you specify only one radix, you can omit the parentheses. If you specify /PARALLEL, /MODE is ignored.

#### /NUMBER (default)

#### /NONUMBER

Includes line numbers in the listing of differences.

**/OUTPUT[=*file-spec*]**

Specifies an output file to receive the list of differences. If **/OUTPUT** is omitted, the list is written to the terminal. *File-spec* defaults to that of the first input file with a file type of DIF. No wildcards are allowed.

**/PARALLEL[=*n*]**

Lists the records with differences side by side. *N* specifies the number of matched records to merge after each unmatched record. *N* must be less than or equal to the number specified in **/MATCH**; defaults to 0.

**/SEPARATED[=(*input1-file-spec*,*input2-file-spec*)]**

Lists only the records from the specified file that contain differences. If no files are specified, a separate listing is generated for each file. If only one file is specified, you can omit the parentheses. To specify *input1-file-spec*, use either the first input file specified as the DIFFERENCES parameter or the keyword MASTER. To specify *input2-file-spec*, use either the second input file specified as the DIFFERENCES parameter or the keyword REVISION.

**/WIDTH=*n***

Specifies the width of the lines in the output file. The default is 132 characters. If output is written to the terminal, **/WIDTH** is ignored and the terminal line width is used.

**/WINDOW=*n***

Searches the number of records specified (*n*) before a record is declared as unmatched. By default, DIFFERENCES searches to the ends of both input files.

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

Requires READ access to the directories or sufficient privilege to override the protection to obtain information other than the file name.

Displays the names and attributes of files in a directory or directories.

**PARAMETERS**

**file-spec**

Specification of file being listed. Wildcard characters are allowed. If no file is specified, all files in your default directory are listed. If just a directory name is specified, all files in that directory are listed. If the file type and version number are omitted, all versions of all file types of the specified file are listed. If the version number is omitted, all versions of the specified file are listed. The plus sign can be used in place of the comma between file specifications.

## **DCL-50     DCL Commands**

### **DIRECTORY**

#### **QUALIFIERS**

##### **/ACL**

Displays the access control list (ACL) for each file. The /ACL qualifier overrides the /COLUMNS qualifier.

##### **/BACKUP**

##### **/CREATED (default)**

##### **/EXPIRED**

##### **/MODIFIED**

Selects files for the directory operation according to the dates of their most recent backups, their creation dates, their expiration dates, or the dates of their last modifications. Relevant only with the /BEFORE and /SINCE qualifiers.

##### **/BEFORE[=time]**

Lists only those files with dates that precede the specified time. You can specify time as an absolute time, a combination of absolute and delta times, or one of the following keywords: TODAY (default), TOMORROW, or YESTERDAY. Specified with /BACKUP, /CREATED (default), /EXPIRED, or /MODIFIED.

##### **/BRIEF (default)**

Displays only a file's name, type, and version number. You can use the /ACL, /DATE, /FILE\_ID, /NOHEADING, /OWNER, /PROTECTION, /SECURITY, and /SIZE qualifiers to expand a brief display.

##### **/BY\_OWNER[=uic]**

Displays only those files with the specified user identification code. The default UIC is that of the current process.

##### **/COLUMNS=columns**

Specifies the number of columns in a brief display. The default is four. The number of columns is restricted by the value of the /WIDTH qualifier. The /COLUMNS qualifier is incompatible with /ACL and /FULL.

##### **/CREATED**

See /BACKUP.

##### **/DATE[=option]**

##### **/NODATE (default)**

Expands the display to include dates. Possible options are:

ALL	Creation, expiration, backup, and last modification dates
BACKUP	Last backup date
CREATED	Creation date
EXPIRED	Expiration date
MODIFIED	Last modification date

**/EXCLUDE=(file-spec,...)**

Excludes files from the display. The file specification can include a directory but not a device name. Wildcard characters are supported for file specifications. However, you cannot use relative version numbers to exclude a specific version.

**/EXPIRED**

See /BACKUP.

**/FILE\_ID**

Displays the file's identification number (FID). By default, a file's identification is not displayed unless the /FULL qualifier is specified.

**/FULL**

Displays the following information: file specification, size (blocks used, blocks allocated), creation date, last backup date, last modification date, expiration date, owner UIC, protection, file identification number (FID), file organization (sequential or indexed), file attributes, record attributes, record format, and access control list (ACL).

**/GRAND\_TOTAL**

Displays only the totals for all specified files and directories.

**/HEADING (default)**

**/NOHEADING**

Prints headers consisting of the device and directory in which the files reside. When /NOHEADING is specified, the display is in single-column format and the device and directory information appears with each file name. The /NOHEADING qualifier overrides /COLUMNS.

**/MODIFIED**

See /BACKUP.

**/OUTPUT[=file-spec]**

**/NOOUTPUT**

Specifies the name of a file to which the directory display is written; by default, the display is written to the current SYS\$OUTPUT device. Wildcard characters are not allowed.

## **DCL-52    DCL Commands**

### **DIRECTORY**

#### **/OWNER**

##### **/NOOWNER (default)**

Displays the owner UIC of the file.

#### **/PRINTER**

Puts the display in a file and queues the file to SYS\$PRINT for printing. The name of the file is as specified in the /OUTPUT qualifier (or, if /OUTPUT is not specified, a temporary file named DIRECTORY.LIS which is queued for printing and then deleted).

#### **/PROTECTION**

##### **/NOPROTECTION (default)**

Displays the protection on the file.

#### **/SECURITY**

Displays information about file security (equivalent to the /ACL, /OWNER, and /PROTECTION qualifiers together).

#### **/SELECT=SIZE=(MINIMUM=n,MAXIMUM=n)**

Selects files for display according to size.

MAXIMUM=n    Displays files that have fewer blocks than the value of *n*, which defaults to 1073741823

MINIMUM=n    Displays files that have more blocks than the value of *n*, which defaults to 0

#### **/SINCE[=time]**

Selects for display only those files dated after the specified date. You can specify time as an absolute time, a combination of absolute and delta times, or one of the following keywords: TODAY (default), TOMORROW, or YESTERDAY. Specified with /BACKUP, /CREATED (default), /EXPIRED, or /MODIFIED.

#### **/SIZE[=option]**

##### **/NOSIZE (default)**

Displays the size of each file. If you omit *option*, the default lists the file size in blocks used (USED). Specify *option* as one of the following:

ALL                Lists blocks allocated and blocks used

ALLOCATION        Lists blocks allocated

USED              Lists blocks used

#### **/TOTAL**

Displays only the directory name and total number of files.



**/TRAILING (default)**  
**/NOTRAILING**

Displays summary information: the number of files listed, the total number of blocks used, and the total number of blocks allocated. If more than one directory is listed, the summary includes the total number of directories, total number of blocks used, and total number of blocks allocated. The /SIZE and /FULL qualifiers determine more precisely what summary information is included.

**/VERSIONS=versions**

Specifies the number of versions of a file to describe and defaults to all versions.

**/WIDTH=(keyword,...)**

Formats the width of the display. Possible keywords are:

DISPLAY=n	Specifies the total width of the display as an integer in the range 1 through 256 and defaults to 0 (setting the display width to the terminal width)
FILENAME=n	Specifies the width of the file name field; defaults to 19
OWNER=n	Specifies the width of the owner field; defaults to 20
SIZE=n	Specifies the width of the size field; defaults to 6

**DISCONNECT**

Breaks the connection between a physical terminal and a virtual terminal. After the physical terminal is disconnected, both the virtual terminal and the process using it remain on the system.

**QUALIFIER**

**/CONTINUE**  
**/NOCONTINUE (default)**

Permits an interrupted image to continue after the disconnect takes place.

**DISMOUNT device-name[:]**

Requires GRPNAM or SYSNAM privilege to dismount group or system volumes, respectively.

Closes a mounted disk or magnetic tape volume for further processing and deassigns the logical name associated with the device. If the volume is mounted with the /SHARE qualifier, its logical name is deassigned but the volume remains mounted until all processes using it dismount it or terminate. Note that all open files on the volume must be closed before the actual dismount can be done. Note, also, that the

## **DCL-54    DCL Commands**

### **DISMOUNT**

file system cannot dismount a volume while any known file lists associated with it contain entries.

#### **PARAMETERS**

##### **device-name[:]**

Name of the device containing the volume — either a logical name or a physical name. If a physical name is specified, the controller defaults to A and the unit defaults to 0.

#### **QUALIFIERS**

##### **/ABORT**

Requires VOLPRO privilege if you are not the owner of the volume.

Specifies that the volume is to be dismounted, regardless of who mounted it. The primary purpose of the /ABORT qualifier is to terminate mount verification. DISMOUNT/ABORT also cancels any outstanding I/O requests. If the volume was mounted with the /SHARE qualifier, /ABORT causes the volume to be dismounted for all users.

##### **/UNIT**

Dismounts only the volume of a volume set on the specified device. By default, all volumes in a set are dismounted.

**NOTE:** Avoid dismounting the root volume of a volume set since it contains the master file directory (MFD).

##### **/UNLOAD (default)**

##### **/NOUNLOAD**

Unloads the device on which the volume is mounted. If you specify /NOUNLOAD, the device remains in a ready state.

#### **DUMP file-spec**

Displays the contents of a file, disk volume, or magnetic tape volume in decimal, hexadecimal, or octal format, as well as the ASCII conversion.

#### **PARAMETERS**

##### **file-spec**

Specification of the file or name of the device being dumped.

## QUALIFIERS

### **/ALLOCATED**

Includes in the dump all blocks allocated to the file. (By default, the dump does not include blocks following the end-of-file.) /ALLOCATED and /RECORDS are mutually exclusive.

### **/BLOCKS[=START:n,END:n,COUNT:n]**

Dumps the specified blocks. Block numbers are specified as integers relative to the beginning of the file. Typically, blocks are numbered beginning with 1. If a disk device is mounted /FOREIGN, blocks are numbered beginning with 0. START specifies the number of the first block to be dumped; the default is the first block. END specifies the number of the last block to be dumped; the default is the last block or the end-of-file block, depending on the /ALLOCATED qualifier. COUNT specifies the number of files to be dumped; COUNT provides an alternative to END. /BLOCKS and /RECORDS are mutually exclusive.

### **/BYTE**

Formats the dump in bytes. /BYTE, /LONGWORD, and /WORD are mutually exclusive.

### **/DECIMAL**

Dumps the file in decimal radix. /DECIMAL, /HEXADECIMAL, and /OCTAL are mutually exclusive.

### **/FILE\_HEADER**

Dumps each data block that is a valid Files-11 header in Files-11 header format rather than the selected radix and length.

### **/FORMATTED (default)**

### **/UNFORMATTED**

Dumps the file header in Files-11 format; /UNFORMATTED dumps the file header in octal format. This qualifier is useful only when /HEADER is specified.

### **/HEADER**

Dumps the file header and access control list. To dump only the file header, also specify /BLOCK=(COUNT:0). /HEADER is invalid for devices mounted /FOREIGN.

### **/HEXADECIMAL (default)**

Dumps the file in hexadecimal radix. /DECIMAL, /HEXADECIMAL, and /OCTAL are mutually exclusive.

### **/LONGWORD (default)**

Formats the dump in longwords. /BYTE, /LONGWORD, and /WORD are mutually exclusive.

## **DCL-56    DCL Commands**

### **DUMP**

#### **/NUMBER[=n]**

Specifies how byte offsets are assigned to the lines of output. If you specify /NUMBER, the byte offsets increase continuously through the dump, beginning with n; if you omit /NUMBER, the first byte offset is 0. By default, the byte offset is reset to 0 at the beginning of each block or record.

#### **/OCTAL**

Dumps the file in octal radix. /DECIMAL, /HEXADECIMAL, and /OCTAL are mutually exclusive.

#### **/OUTPUT[=file-spec]**

Specifies the output file. The default is the file name of the file being dumped and the file type DMP. If this qualifier is not specified, the dump goes to SYS\$OUTPUT. No wildcard characters are allowed. /OUTPUT and /PRINTER are mutually exclusive.

#### **/PRINTER**

Queues the dump to SYS\$PRINT in a file named with the file name of the file being dumped and the file type DMP. If this qualifier is not specified, the dump goes to SYS\$OUTPUT. No wildcard characters are allowed. /OUTPUT and /PRINTER are mutually exclusive.

#### **/RECORDS[=START:n,END:n,COUNT:n]**

Dumps the file a record at a time rather than a block at a time. Records are numbered beginning with 1. START specifies the number of the first record to be dumped; the default is the first record. END specifies the number of the last record to be dumped; the default is the last record. COUNT specifies the number of records to be dumped; provides an alternative to END. If you specify /RECORDS, you cannot specify /ALLOCATED or /BLOCKS.

#### **/WORD**

Formats the dump in words. /BYTE, /LONGWORD, and /WORD are mutually exclusive.

### **EDIT/ACL**

See Appendix ACL.

### **EDIT/EDT**

See Appendix EDT.

## **EDIT/TPU [file-spec]**

Invokes the VAX Text Processing Utility (VAXTPU) with the Extensible VAX Editor (EVE) interface. To invoke VAXTPU with the EDT Keypad Emulator interface, define the logical TPUSECINI to point to the section file for that interface as follows:

```
$ DEFINE TPUSECINI EDTSECINI
```

### **PARAMETERS**

#### **file-spec**

Specification of the file being edited. If you do not supply a file specification and you modify a buffer, VAXTPU prompts you for a file specification when you leave the utility.

### **QUALIFIERS**

#### **/COMMAND[=file-spec] (default)**

##### **/NOCOMMAND**

Determines whether VAXTPU executes a command file before the editing session begins. File-spec defaults to TPUINI.TPU in your default directory. You can override the default, by defining the logical name TPUINI to point to a different file, or by specifying a different file-spec after /COMMAND.

#### **/CREATE (default)**

##### **/NOCREATE**

Determines whether VAXTPU provides a buffer in which to create a new file when the specified input file is not found. The interface layered on VAXTPU is responsible for processing this qualifier.

#### **/DISPLAY[=file-spec] (default)**

##### **/NODISPLAY**

Determines whether a VAXTPU session is run from a supported terminal and uses terminal functions such as the screen display and keyboard. By default your VAXTPU session is run with the screen management file TPU\$CCTSHR.EXE, for terminals that respond to ANSI control functions and that operate in ANSI mode. Use /NODISPLAY when running VAXTPU procedure in batch, or when using VAXTPU on an unsupported terminal.

#### **/JOURNAL[=file-spec] (default)**

##### **/NOJOURNAL**

Determines whether VAXTPU keeps a journal file during an editing session. The default is /JOURNAL=filename.TJL, where filename is the name of the file being edited. If no file name is specified with EDIT/TPU, file-spec defaults to TPU.TJL. The interface layered on VAXTPU is responsible for processing this qualifier.



## **DCL-58    DCL Commands**

### **EDIT/TPU**

#### **/OUTPUT[=file-spec] (default)**

##### **/NOOUTPUT**

Determines whether VAXTPU creates an output file at the end of an editing session. The default is /OUTPUT=input-file-spec, where the file name and the file type remain unchanged and the version number is one higher than the highest existing version of the input file. The interface layered on VAXTPU is responsible for processing this qualifier.

#### **/READ\_ONLY**

##### **/NOREAD\_ONLY (default)**

Determines whether both an output file and a journal file are created. With /NOREAD\_ONLY, VAXTPU maintains a journal file (in case you modify the main buffer and an interruption occurs), and creates an output file when the command EXIT is issued. Use the qualifier /READ\_ONLY when you want the main buffer set to NO\_WRITE, for example, when you want to look at a file but you do not intend to make any changes to it. When you use the qualifier /READ\_ONLY, enter the command QUIT to leave VAXTPU. You can use the command WRITE FILE in EVE, or the built-in procedure WRITE\_FILE in the EDT Keypad Emulator to write out a buffer that is set to NO\_WRITE.

#### **/RECOVER**

##### **/NORECOVER (default)**

Determines whether a journal file is executed before the editing session begins. If the name of the journal file is different from that of the input file, you must use both /RECOVER and /JOURNAL and you must specify the name of the journal file with the qualifier /JOURNAL.

#### **/SECTION[=file-spec] (default)**

##### **/NOSECTION**

Determines whether VAXTPU reads an initialization file that is stored in binary form. A section file must have been compiled by running the source code version through VAXTPU and then using the built-in procedure SAVE. File-spec defaults to SYS\$LIBRARY:TPUSECINI.TPU\$SECTION, the section file that creates the EVE editing interface. You can specify a different file for initialization purposes by defining the logical name TPUSECINI to point to a section file. This is the preferred method. However, you can also supply a full file specification for /SECTION.

## **EOD**

Terminates a data line that begins with a dollar sign, or terminates an input file if more than one input file is contained in the command stream without intervening commands.



## **EXAMINE location[:location]**

Requires user mode READ and WRITE access to the virtual memory location.

Displays the specified virtual memory location, maintaining a pointer at that location.

### **PARAMETERS**

#### **location**

A virtual address or a range of virtual addresses (where the second address is larger than the first). A location can be any valid arithmetic expression containing arithmetic or logical operators or previously assigned symbols. Radix qualifiers determine the radix in which the address is interpreted; hexadecimal is the initial default radix. Symbol names are always interpreted in the radix in which they were defined. The radix operators %X, %D, or %O can precede the location. A hexadecimal value must begin with a number (or be preceded by %X).

### **QUALIFIERS**

#### **/ASCII**

Displays the data in ASCII; uses hexadecimal as the default radix for numeric literals that are specified on the command line. Binary values that do not have ASCII equivalents are displayed as periods.

#### **/BYTE**

#### **/LONGWORD**

#### **/WORD**

Displays the data in bytes, longwords, or words. The initial default is longwords.

#### **/DECIMAL**

#### **/HEXADECIMAL**

#### **/OCTAL**

Displays the data in decimal, hexadecimal, or octal. The initial default is hexadecimal.

#### **/HEXADECIMAL**

See /DECIMAL.

#### **/LONGWORD**

See /BYTE.

#### **/OCTAL**

See /DECIMAL.

#### **/WORD**

See /BYTE.

**EXIT [status-code]**

Terminates processing of a command procedure and returns control to the next higher command level — either an invoking command procedure or DCL. The EXIT command also terminates an image normally after CTRL/Y is typed (executing another image has the same effect).

**PARAMETERS****status-code**

Longword (integer) value giving the exit status of the image. (This value is assigned to the global symbol \$STATUS and the lower three bits determine the value of the global symbol \$SEVERITY.)

**GOSUB label**

Transfers control to a labeled subroutine in a command procedure without creating a new procedure level. The RETURN command terminates the GOSUB subroutine.

**PARAMETER****label**

A valid label (1 through 255 alphanumeric characters, terminated by a colon, first item on the line) in the command procedure. If two or more labels are identical, control passes to the nearest label preceding the GOSUB command, or to the nearest label following the GOSUB command if no duplicate label precedes the command.

**GOTO label**

Transfers control to a labeled statement in a command procedure.

**PARAMETERS****label**

A valid label (1 through 255 alphanumeric characters, terminated by a colon, first item on the line) in the command procedure. If two or more labels are identical, control passes to the nearest label preceding the GOTO command, or to the nearest label following the GOTO command if no duplicate label precedes the command.

**HELP [topic[subtopic]...]**

Displays information concerning use of the system, including formats and explanations of commands, parameters, and qualifiers.

## PARAMETERS

### **topic**

Name of the topic. If only a topic name is specified, information concerning the topic is displayed along with a list of subtopics at the next level. If topic... is specified, information concerning the topic and all subtopics is displayed. If topic \* is specified, information concerning all subtopics is displayed. An asterisk or percent sign within a topic name serves as a wildcard. If you type just HELP or if the topic contains subtopics, you are prompted. Type the subtopic name to get help on the subtopic. Type a question mark to display the topic information again. Press RETURN to back up a level in the topic hierarchy. Press CTRL/Z to exit directly to DCL command level.

### **subtopic**

Name of a subtopic. Information concerning the lowest subtopic specified is displayed along with a list of any further subtopics. If subtopic... is specified, information concerning the subtopic and all lower subtopics is displayed. If subtopic \* is specified, information concerning all lower subtopics is displayed. An asterisk or percent sign within a subtopic name serves as a wildcard.

## QUALIFIERS

### **/INSTRUCTIONS (default)**

#### **/NOINSTRUCTIONS**

Displays an explanation of the HELP command along with the list of topics (if no topic is specified).

### **/LIBLIST (default)**

#### **/NOLIBLIST**

Displays any auxiliary help libraries.

### **/LIBRARY=file-spec (default)**

#### **/NOLIBRARY**

Names the main help library. Defaults to SYS\$HELP (which is normally the logical name for [SYSHLP]HELPLIB.HLB on the system disk). The file type defaults to HLB. No wildcards are allowed.

### **/OUTPUT[=file-spec]**

#### **/NOOUTPUT**

Specifies an output file to which the information is written; by default, the information is written to the current SYS\$OUTPUT device. No wildcards are allowed.

### **/PAGE (default)**

#### **/NOPAGE**

Stops the display when the screen is full. You must press RETURN to continue.

## **DCL-62    DCL Commands**

### **HELP**

**/PROMPT (default)**

**/NOPROMPT**

Permits you to solicit further information interactively.

**/USERLIBRARY[=(level,...) (default)**

**/NOUSERLIBRARY**

Names the levels of search for information in auxiliary libraries.

PROCESS	Libraries defined at process level
GROUP	Libraries defined at group level
SYSTEM	Libraries defined at system level
ALL	All libraries (default)
NONE	No libraries (same as /NOUSERLIBRARY)

Auxiliary help libraries are libraries defined with the logical names HLP\$LIBRARY, HLP\$LIBRARY\_1, HLP\$LIBRARY\_2, and so on. Libraries are searched for information in this order: current library, main library (if not current), libraries defined at process level, libraries defined at group level, and libraries defined at system level. The default is /USERLIBRARY=ALL.

## **IF expression THEN command**

Tests the value of the specified expression and executes the commands in the THEN clause if the expression is true.

**PARAMETERS**

**expression**

An expression that evaluates to a logical or numeric value.

**command**

Any DCL command.

## **INITIALIZE device-name volume-label**

Formats a disk or magnetic tape volume and writes a label on the volume. At the end of initialization, the disk is empty except for the system files containing the structure information. All former contents of the disk are lost.

**PARAMETERS**

**device-name**

Name of the device on which the volume is physically mounted.

**volume-label**

Specifies the identification to be encoded on the volume. For a disk volume, you can specify a maximum of 12 alphanumeric characters; for a magnetic tape volume, you can specify a maximum of 6 alphanumeric characters. Letters are automatically changed to uppercase. Nonalphanumeric characters are not allowed in the volume-label specification on disk.

**QUALIFIERS**

**/ACCESSED=number-of-directories**

Requires OPER privilege.

Specifies the number of directories to be maintained in system space for ready access as an integer in the range 0 through 255. Defaults to 3.

**/BADBLOCKS=(area,...)**

Specifies faulty areas on a volume so that no data will be written to them. Possible formats for area are:

lbn[:count]	Logical block number of the first block and optionally a block count beginning with the first block
sec.trk.cyl[:cnt]	Sector, track, and cylinder of the first block, and optionally a block count beginning with the first block

**/CLUSTER\_SIZE=number-of-blocks**

Defines the minimum allocation unit in blocks. The maximum size is 1/100 of the volume size. The minimum size is 255\*4096 divided into the disk size in blocks. Structure Level 2 disks smaller than 50,000 blocks have a default value of 1. Structure Level 1 disks must always have a cluster size of 1.

**/DATA\_CHECK[=(keyword,...)]**

Checks all read and/or write operations on the disk. By default, no data checks are made. Specify one or both keywords:

READ	Checks all read operations
WRITE	Checks all write operations; default if only /DATA_CHECK is specified

**/DENSITY=density-value**

The /DENSITY qualifier is not applicable to the TK50 tape device.

For floppy disk volumes that are to be initialized on RX02 dual-density disk drives, specifies the density at which the floppy disk is to be formatted.

For magnetic tape volumes, specifies the density in bytes per inch (bpi) at which the magnetic tape is to be written.

RX02 dual-density disk drives allow floppy disks to be initialized at single or double density. To specify single-density formatting of a floppy disk, specify the density value SINGLE. To specify double-density formatting of a floppy disk, specify the density value DOUBLE.



## **DCL-64    DCL Commands**

### **INITIALIZE**

If you do not specify a density value for a floppy disk being initialized on an RX02 drive, the system leaves the volume at the density to which the volume was last formatted. Floppy disks purchased from DIGITAL are formatted in single density.

For magnetic tape volumes, the density value specified can be 800 bpi, 1600 bpi, or 6250 bpi, as long as the density is supported by the magnetic tape drive. If you do not specify a density value for a blank magnetic tape, the system uses a default density of the highest value allowed by the tape drive. If the drive allows 6250, 1600, and 800 bpi operation, the default density is 6250. If the drive allows only 1600 and 800 bpi operation then the default density is 1600. If you do not specify a density value for a magnetic tape that has been previously written, the system uses the density of the first record on the volume. The magnetic tape density will not default on an unusually short record.

#### **/DIRECTORIES=number-of-entries**

Allocates the specified number of entries for user directories as an integer in the range 16 through 16000. The default is 16.

#### **/ERASE**

##### **/NOERASE (default)**

Physically destroys deleted data (by writing over it).

#### **/EXTENSION=n**

Affects Files-11 Structure Level 1 disks ONLY

Specifies, for disk volumes, the number of blocks to use as a default extension size for all files on the volume. The value *n* can range from 0 through 65,535. The default is 5.

#### **/FILE\_PROTECTION=code**

Affects Files-11 Structure Level 1 disks ONLY

Defines, for disk volumes, the default protection to be applied to all files on the volume.

#### **/GROUP**

Defines a group volume. The /GROUP qualifier applies protection of RWED to all ownership categories unless /GROUP is specified with /NOSHARE, in which case the volume protection is RWED for all but the world category. The owner UIC of the volume defaults to your group number and a member number of 0.

#### **/HEADERS=number-of-headers**

Specifies the number of file headers to be allocated for the index file. The minimum and default value is 16. The maximum is the value set with the /MAXIMUM\_FILES qualifier.



**/HIGHWATER (default)**  
**/NOHIGHWATER**

Sets the file highwater mark (FHM) volume attribute, which guarantees that a user cannot read data that he has not written. You cannot specify /NOHIGHWATER for magnetic tape.

**/INDEX=keyword**

Specifies the location of the index file for the volume's directory structure. Possible keywords are:

BEGINNING	Beginning of the volume
MIDDLE	Middle of the volume (default)
END	End of the volume
BLOCK:n	Beginning of the logical block specified by <i>n</i>

**/LABEL=option**

Defines characteristics for the magnetic tape volume label, as directed by the included option. The available options are as follows:

**OWNER\_IDENTIFIER:"(14 ANSI characters)"**

Allows you to specify the Owner Identifier field in the volume label. The field specified can accept up to 14 ANSI characters.

**VOLUME\_ACCESSIBILITY:"character"**

Specifies the character to be written in the volume-accessibility field of the MicroVMS ANSI volume label VOL1 on an ANSI magnetic tape. The character may be any valid ANSI "a" character. This set of characters includes numeric characters, uppercase letters, and any one of the following nonalphanumeric characters:

! " % ' ( ) \* + , - . / : ; < = > ?

By default, MicroVMS provides a routine that checks this field in the following manner. If the magnetic tape was created on a version of MicroVMS that conforms to Version 3 of ANSI, then this option must be used to override any character other than an ASCII space. If a MicroVMS protection is specified and the magnetic tape conforms to an ANSI standard that is later than that of Version 3, then this option must be used to override any character other than an ASCII 1. If you specify any character other than the default character, you must specify the /OVERRIDE=ACCESSIBILITY qualifier on the INITIALIZE and MOUNT commands in order to access the magnetic tape.

## DCL-66    DCL Commands

### INITIALIZE

#### **/MAXIMUM\_FILES=number-of-files**

Restricts the maximum number of files that the volume can contain. The /MAXIMUM\_FILES qualifier overrides the default value, which is calculated as follows:

```
volume size in blocks
-----
(cluster factor + 1) * 2
```

The maximum size you can specify for any volume is:

```
volume size in blocks
-----
(cluster factor + 1)
```

The minimum value is 0. Note that the maximum can be increased only by reinitializing the volume.

#### **/OVERRIDE=(option,...)**

Requests the INITIALIZE command to ignore data on a magnetic tape volume that protects it from being overwritten. You may specify one of the following options:

ACCESSIBILITY

(For magnetic tapes only.) If the installation allows, this option overrides any character in the Accessibility Field of the volume. The necessity of this option is defined by the installation. That is, each installation has the option of specifying a routine that the magnetic tape file system will use to process this field. By default, MicroVMS provides a routine that checks this field in the following manner. If the magnetic tape was created on a version of MicroVMS that conforms to Version 3 of ANSI, then this option must be used to override any character other than an ASCII space. If a MicroVMS protection is specified and the magnetic tape conforms to an ANSI standard that is later than Version 3, then this option must be used to override any character other than an ASCII 1. To use the ACCESSIBILITY option, you must have the user privilege VOLPRO or be the owner of the volume.

EXPIRATION

(For magnetic tapes only). Allows you to write to a tape that has not yet reached its expiration date. You must have the user privilege VOLPRO to override volume protection, or your UIC must match the UIC written on the volume.

OWNER\_IDENTIFIER

Allows you to override the processing of the Owner Identifier field of the volume label.

If you specify only one option, you may omit the parentheses.

In order to initialize a volume that was initialized previously with the /PROTECTION qualifier, your UIC must match the UIC written on the volume or you must have VOLPRO privilege.

**/OWNER\_UIC=uic**

Specifies an owner UIC for the volume. The default is your default UIC.

**/PROTECTION=(ownership[:access],...)**

Applies the specified protection to the volume. The default is your default protection. The ownership categories are SYSTEM, OWNER, GROUP, WORLD; the access categories are R (read), W (write), E (create), and D (delete).

**/SHARE (default)**

**/NOSHARE**

Permits all categories of access by all categories of ownership. The /NOSHARE qualifier denies access to group (unless /GROUP is also specified) and world processes.

**/STRUCTURE=level**

Specifies whether the volume should be formatted in Files-11 Structure Level 1 or Structure Level 2 (the default). Level 1 is incompatible with the /DATA\_CHECK and /CLUSTER\_SIZE qualifiers.

**/SYSTEM**

Requires a system UIC or SYSPRV privilege.

Defines a system volume. The owner UIC defaults to [1,1]. Protection defaults to complete access by all ownership categories, except that only system processes can create top-level directories.

**/USER\_NAME=name**

Specifies a user name to be associated with the volume. The name must be 1 to 12 alphanumeric characters. The default is your user name.

**/VERIFIED**

**/NOVERIFIED**

Indicates whether the disk has bad block data on it. Use the /NOVERIFIED qualifier to ignore bad block data on the disk. The default is /VERIFIED for disks with 4096 blocks or more and /NOVERIFIED for disks with less than 4096 blocks.

**/WINDOWS=number-of-pointers**

Specifies the number of mapping pointers (used to access data in the file) to be allocated for file windows. The value can be an integer in the range 7 through 80. The default is 7.

**DCL-68    DCL Commands**  
**INITIALIZE/QUEUE**

**INITIALIZE/QUEUE queue-name**

Requires OPER privilege.

Creates a print or batch queue and assigns it a name and attributes.

**PARAMETERS**

**queue-name**

User-defined name of the queue. *Queue-name* may be up to 31 alphanumeric characters.

**QUALIFIERS**

**/BASE\_PRIORITY=priority**

Specifies the process base priority at which jobs are initiated from a batch queue or the base priority of the symbiont process for a printer, terminal, or server queue. The value can be an integer in the range 0 through 15. The base priority defaults to the same priority as the base priority established by DEFPRI at system generation (usually 4).

**/BATCH**

**/NOBATCH (default)**

Specifies that the queue is a batch queue. If you specify /NOBATCH or omit this qualifier, the queue is assumed to be a printer queue.

**/BLOCK\_LIMIT=(*[lower,upper]*)**

**/NOBLOCK\_LIMIT (default)**

Restricts the size of print jobs that can be executed on a printer, terminal, or server queue. The lower parameter specifies the minimum number of blocks that will be accepted by the queue for a print job. The upper parameter specifies the maximum number of blocks that will be accepted by the queue for a print job. If a job contains fewer blocks than the number specified by the lower parameter or more blocks than the number specified by the upper parameter, the job remains pending until the block limit for the queue is changed. To specify only the lower parameter, you must use two sets of quotation marks (""") in place of the upper specifier.

**/CHARACTERISTICS=(*characteristic,...*)**

**/NOCHARACTERISTICS (default)**

Specifies one or more characteristics for processing jobs on the queue. (Use the SHOW QUEUE/CHARACTERISTIC command to display the available characteristics.) A queue must have all the characteristics specified for a job or the job remains pending.

**/CPUDEFAULT=time**

Specifies the default CPU time limit for batch jobs. Time can be specified as a delta time, 0, NONE (the default), or INFINITE. Both the value 0 and the keyword INFINITE allow unlimited CPU time (subject to the restrictions imposed by the /CPUMAXIMUM qualifier or the user authorization file).

**/CPUMAXIMUM=time**

Specifies the maximum CPU time limit for batch jobs. The /CPUMAXIMUM qualifier overrides the time limit specified in the user authorization file (UAF). Time can be specified as a delta time, 0, NONE (the default), or INFINITE. Both the value 0 and the keyword INFINITE allow unlimited CPU time.

**/DEFAULT=(option,...)**

**/NODEFAULT**

Establishes default options for the PRINT command. The /DEFAULT qualifier can not be used with the /GENERIC qualifier. Possible options are:

[NO]BURST=[keyword]

Specifies where to print burst pages (flag pages that are printed over the paper's perforations for easy identification of individual files in a print job). The keyword ALL places burst pages before each printed file in the job. The keyword ONE places a burst page before the first printed file in the job.

[NO]FEED

Specifies whether a form feed is automatically inserted at the end of a page. (The default is FEED.)

[NO]FLAG=[keyword]

Specifies where to print flag pages (containing the job entry number, the name of the user submitting the job, and so on). The keyword ALL places flag pages before each printed file in the job. The keyword ONE places a flag page before the first printed file in the job.

FORM=type

Specifies the default form for a printer, terminal, or server queue. If a job is not submitted with an explicit form definition, then this form will be used to process the job. The systemwide default form, form=0, is the default value for this keyword. See also /FORM\_MOUNTED.

[NO]TRAILER=[keyword]

Specifies where to print trailer pages. The keyword ALL places trailer pages after each printed file in the job. The keyword ONE places a trailer page after the last printed file in the job.

If you specify any of the keywords BURST, FLAG, or TRAILER without specifying a value, the value ALL is used by default.



## **DCL-70    DCL Commands**

### **INITIALIZE/QUEUE**

#### **/DISABLE\_SWAPPING**

#### **/NODISABLE\_SWAPPING (default)**

Controls whether batch jobs executed from a queue can be swapped in and out of memory.

#### **/ENABLE\_GENERIC (default)**

#### **/NOENABLE\_GENERIC**

Determines whether or not a queue will accept files from a generic queue of the same type (where the generic queue was initialized without a queue name specified with the /GENERIC qualifier).

#### **/FORM\_MOUNTED=type**

Specifies the form type for a printer, terminal, or server queue. If the stock of the mounted form is not identical to the stock of the default form, as indicated by the DCL command qualifier /DEFAULT=FORM=type, then all jobs submitted to this queue without an explicit form definition will enter a pending state. If a job is submitted with an explicit form and the stock of the explicit form is not identical to the stock of the mounted form, then the job will enter a pending state. In both cases, the pending state will be maintained until the stock of the mounted form of the queue is identical to the stock of the form associated with the job. The /FORM\_MOUNTED qualifier can not be used with the /GENERIC qualifier.

#### **/GENERIC[=(queue-name,...)]**

#### **/NOGENERIC (default)**

Specifies that the queue is generic and that the jobs placed in it can be moved to compatible execution queues for processing. If you specify one or more names with /GENERIC, jobs can be moved only to the specified queues. The target execution queues may be initialized with the /NOENABLE\_GENERIC qualifier to disable them from accepting jobs from the generic queues that do not specify an explicit execution queue list. If you do not specify a queue name with /GENERIC, jobs can be moved to any execution queue (with /ENABLE\_GENERIC in effect) that is the same type (batch, printer, terminal, or server) as the generic queue. By default, a generic queue is a print queue. Use the appropriate qualifier (/BATCH, /PROCESSOR, /TERMINAL) to override the default. The /GENERIC qualifier is incompatible with the /DEFAULT, /FORM\_MOUNTED, and /SEPARATE qualifiers. (The generic and execution queues must be initialized with matching /BATCH and /PROCESSOR queues.)

The /BATCH qualifier determines that an execution queue is a batch queue. The symbiont process determines whether queues are printer, terminal, or server queues; the standard symbiont sets this characteristic depending upon whether the output device is a printer or a terminal.



**/JOB\_LIMIT=number-of-jobs**

Specifies the number of batch jobs that can be executed concurrently from the queue. Defaults to 1.

**/LIBRARY=filename (default)**

**/NOLIBRARY**

Assigns a device control library (containing escape sequence modules) for programmable printers. (The /LIBRARY qualifier can be used to specify an alternate device control library when used to initialize an output queue.) The default library is SYS\$LIBRARY:SYSDEVCTL.TLB. You can specify only a file name. The library must be in SYS\$LIBRARY and the file type must be TLB.

**/ON=device[:]**

Specifies the device on which the queue is located. The default *device-name* is the queue name.

**/OWNER\_UIC=uic**

Specifies the user identification code (UIC) of the queue. The default UIC is [1,4].

**/PROCESSOR=filename**

**/NOPROCESSOR**

Specifies a file containing a print symbiont. Specify only a file name. The file must be in SYS\$SYSTEM and must have a type of EXE. The default print symbiont is named PRTSMB. (Used for a generic queue, the /PROCESSOR qualifier specifies that the generic queue can place jobs only on queues that were defined as server queues and that are executing the specified symbiont image.)

**/PROTECTION=(ownership[:access],...)**

Specifies the protection of the queue. Ownership categories are: SYSTEM, OWNER, GROUP, WORLD; each category can be abbreviated to its first character. Access categories are: R (read), W (write), E (execute), or D (delete); a null access specification means no access. The default protection is: (SYSTEM:E, OWNER:D, GROUP:R, WORLD:W).

**/RECORD\_BLOCKING (default)**

**/NORECORD\_BLOCKING**

Determines whether the symbiont can concatenate (or block together) output records for transmission to the output device. If you specify /NORECORD\_BLOCKING, the symbiont is directed to send each formatted record in a separate I/O request to the output device. For the standard MicroVMS print symbiont, record blocking can have a significant performance advantage over single-record mode.

## **DCL-72    DCL Commands**

### **INITIALIZE/QUEUE**

#### **/RETAIN[=keyword]** **/NORETAIN (default)**

Holds jobs in the queue in a completed status after they have executed. Possible keywords are:

ALL (default)	Holds all jobs in the queue after execution
ERROR	Holds in the queue only jobs that complete unsuccessfully

#### **/SCHEDULE=SIZE (default)** **/SCHEDULE=NOSIZE**

Schedules jobs in a print queue on the basis of size so that short jobs print before long jobs. The effect of this qualifier on currently pending jobs is unpredictable.

#### **/SEPARATE=(keyword,...)** **/NOSEPARATE (default)**

Specifies the job separation defaults for a printer or terminal queue. The /SEPARATE qualifier can not be used with the /GENERIC qualifier. Possible keywords are:

[NO]BURST	Prints a burst page (a flag page printed over the paper's perforations for easy identification of individual files) at the beginning of every job.
[NO]FLAG	Prints a flag page (containing the job entry number, the name of the user submitting the job, and so on) at the beginning of every job.
[NO]TRAILER	Prints a trailer page at the end of every job.
[NO]RESET=(m,...)	Specifies a job reset sequence for the queue. The specified modules from the device control library (see /LIBRARY) are used to reset the device each time a job reset occurs.

#### **/START** **/NOSTART (default)**

Starts the queue being initialized by the current INITIALIZE/QUEUE command.

#### **/TERMINAL** **/NOTERMINAL (default)**

Associates a generic queue with a terminal queue (instead of a printer queue) of matching characteristics.

#### **/WSDEFAULT=n**

Defines a working set default for a batch job. The /WSDEFAULT qualifier overrides the working set size specified in the user authorization file. Possible values are: a positive integer in the range 1 through 65,535, 0, or the keyword NONE (the default). A zero or NONE sets the default value to the value specified either in the UAF or by the SUBMIT command (if specified).

When used for an output queue, this qualifier specifies the working set default of a symbiont process for a printer, terminal, or server queue when the symbiont process is created.

**/WSEXTENT=n**

Defines a working set extent for the batch job. The /WSEXTENT qualifier overrides the working set extent in the user authorization file. Possible values are: a positive integer in the range 1 through 65,535, 0, or the keyword NONE (the default). A zero or NONE sets the default value to the value specified either in the UAF or by the SUBMIT command (if specified).

When used for an output queue, this qualifier specifies the working set extent of a symbiont process for a printer, terminal, or server queue when the symbiont process is created.

**/WSQUOTA=n**

Defines a working set page size (working set quota) for the batch job. The /WSQUOTA qualifier overrides the value in the user authorization file. Possible values are: a positive integer in the range 1 through 65,535, 0, or the keyword NONE (the default). A zero or NONE sets the default value to the value specified either in the UAF or by the SUBMIT command (if specified).

When used for an output queue, this qualifier specifies the working set quota of a symbiont process for a printer, terminal, or server queue when the symbiont process is created.

**INQUIRE symbol-name [prompt]**

Reads a value from SYS\$COMMAND (usually the terminal in interactive mode or the next line in the main command procedure) and assigns it to a symbol. (The value must be enclosed in quotation marks ("")) to preserve lowercase characters, multiple spaces, and tabs.)

**PARAMETERS**

**symbol-name**

Name for the symbol. The name must be 1 to 255 alphanumeric characters.

**prompt**

Prompt to be issued in interactive mode. Enclose the prompt in quotation marks ("")) if it contains lowercase characters, punctuation, multiple blanks or tabs, or the at sign (@). The default prompt is the specified symbol name.

## **DCL-74    DCL Commands**

### **INQUIRE**

#### QUALIFIERS

**/GLOBAL**

**/LOCAL (default)**

Makes the symbol global or local to the current command level.

**/PUNCTUATION (default)**

**/NOPUNCTUATION**

Inserts a colon and a space after the prompt.

#### **LIBRARY library-file-spec [input-file-spec,...]**

Creates, modifies, and examines libraries.

#### PARAMETERS

##### **library-file-spec**

Specification of the library file. The file type defaults to OLB for object and shareable image libraries, MLB for macro libraries, HLB for help libraries, and TLB for text libraries. No wildcard characters are allowed.

##### **input-file-spec**

Specification of file containing modules to be added, inserted, or replaced in the library. The file type defaults to OBJ for object files, EXE for shareable image files, MAR for macro files, HLP for help files, and TXT for text files. Wildcard characters are allowed. This parameter is required with /INSERT and /REPLACE, optional with /CREATE, and otherwise invalid. You can specify SYS\$INPUT to input one text module from the terminal: name the module with /MODULE; type the text of the module on lines following the command; terminate the terminal input with CTRL/Z.

#### QUALIFIERS

**/BEFORE[=time]**

Used with the /LIST qualifier to list only those modules with dates (creation, modification, expiration, or backup) that precede the specified time. You can specify time as an absolute time or a combination of absolute and delta times.

**/COMPRESS[=(option,...)]**

Creates a new library file from the current library file, recovering any space left by modules deleted from the library. The name of the new file is as specified in /OUTPUT or defaults to the next version of the current library file. The options are as follows, with the same as defaults the current library file.

BLOCKS:blocks	Number of 512-byte blocks allocated to the new library
GLOBALS:symbols	Maximum number of global symbols allowed in an object or shareable image library
HISTORY:records	Maximum number of history records allowed in the library; if this value is less than the current number in the library, the oldest records are deleted (only applies if KEEP is specified)
KEEP	Saves update history records and additional information in module headers
KEYSIZE:size	Maximum number of characters allowed in any module or global symbol name
MODULES:modules	Maximum number of modules allowed in library

**/CREATE[=(option,...)]**

Creates a library file. If input-file-spec is specified (second parameter), the modules in the input file are added to the library. The options are as follows:

BLOCKS:blocks	Number of blocks allocated to the new library; defaults to 100
GLOBALS:symbols	Maximum number of global symbols allowed in an object or shareable image library; defaults to 128
HISTORY:records	Maximum number of history records allowed in the library; defaults to 20
KEYSIZE:n	Maximum name length of modules or global symbols; defaults to 31 for object and text libraries and to 15 for help modules; maximum size possible is 128
MODULES:modules	Maximum number of modules allowed in the library; defaults to 512 for an object or shareable image library, and 256 for all others

**/CROSS\_REFERENCE[=(option,...)]**

Generates a cross-reference listing file for object libraries. Name the file with /OUTPUT. The options are as follows:

ALL	Equivalent to (MODULE, SYMBOL, VALUE)
MODULE	Lists global symbol references and definitions
NONE	Equivalent to not specifying /CROSS_REFERENCE
SYMBOL	Lists the global symbols by name; the default
VALUE	Lists the global symbols by value; the default

**/DATA=keyword**

Determines whether data is reduced or expanded, depending upon the specified keyword. (Both forms of /DATA perform an implicit /COMPRESS; that is, while reducing or expanding the data format, the librarian also recovers space that had been occupied by modules deleted from the library.)



## **DCL-76    DCL Commands LIBRARY**

**EXPAND**    Expands a reduced library

**REDUCE**    Stores data in reduced format, requiring less disk space but taking longer to access

### **/DELETE=(module,...)**

Deletes the specified modules from the library. Wildcard characters are allowed.

### **/EXTRACT=(module,...)**

Copies the specified modules into a new file. Wildcard characters are allowed. The name of the new file is as specified by /OUTPUT with a default file type of OBJ, EXE, MAR, HLP, or TXT, depending on the type of library; or is, by default, the name of the library file with the default file type.

### **/FULL**

Gives a full description of each module listed by /LIST. When used with /HISTORY/LIST, names the modules inserted or deleted.

### **/GLOBALS (default)**

### **/NOGLOBALS**

Includes global symbols of object modules in the global symbol table.

### **/HELP**

### **/MACRO**

### **/OBJECT (default)**

### **/SHARE**

### **/TEXT**

Identifies the type of library.

### **/HISTORY**

Lists history records. Valid only with /LIST. When used with /LIST/FULL, additionally lists the names of updated modules.

### **/INSERT**

Adds the modules in the specified input file to the library. If a module being added already exists in the library, the new module is not added and an error message is issued.

### **/LIST[=file-spec]**

### **/NOLIST (default)**

Writes header information and lists the contents of the library. The information is written to the specified file, whose file type defaults to LIS. If the file specification is omitted, the information is written to SYS\$OUTPUT. If specified with other operations (for example, /DELETE, /INSERT, or /REPLACE), the listing reflects the status of the library after the other operations are performed.



**/LOG**

**/NOLOG (default)**

Writes a message to SYS\$OUTPUT after each operation (for example, /DELETE, /INSERT, /REMOVE, or /REPLACE) occurs.

**/MACRO**

See /HELP.

**/MODULE=module-name**

Qualifies input-file-spec.

Names an incoming text module. This qualifier applies only to /INSERT and /REPLACE operations on text libraries. The module name defaults to the name of the file. Incompatible with /EXTRACT, /DELETE, and /REMOVE.

**/NAMES**

**/NONAMES (default)**

Lists all the global symbol names and module names in an object module library. This qualifier applies only when /LIST is specified.

**/OBJECT (default)**

See /HELP.

**/ONLY=(module,...)**

Limits a /LIST or /CROSS\_REFERENCE operation to the specified modules. Wildcards are allowed.

**/OUTPUT[=file-spec]**

Specifies the output file specification for a /COMPRESS, /EXTRACT, or /CROSS\_REFERENCE operation. The file type defaults to OLB, MLB, HLB, or TLB for /COMPRESS operations; OBJ, EXE, MAR, HLP, or TXT for /EXTRACT operations; or LIS for /CROSS\_REFERENCE operations. If /OUTPUT is not specified, the default is the name of the library with the default file type. Wildcard characters are not allowed.

**/REMOVE=(symbol-name,...)**

Deletes the specified global symbol names from the object library. Wildcard characters are allowed.

**/REPLACE (default)**

Adds the modules in the specified input file to the library. If a module being added already exists in the library, the new module replaces the existing module. The action specified by /REPLACE is the default librarian operation.

## **DCL-78    DCL Commands LIBRARY**

### **/SELECTIVE\_SEARCH**

Applies only to individual modules in object libraries. When the library is used as input to a link operation, the linker excludes global symbols not referenced by other modules from the resultant image and (if applicable) symbol table.

### **/SHARE**

See /HELP.

### **/SINCE[=time]**

Writes only those modules dated after the specified time when used with the /LIST command. You can specify time as an absolute time or a combination of absolute and delta times.

### **/SQUEEZE (default)**

### **/NOSQUEEZE**

Compresses (deletes trailing blanks, trailing tabs, and comments) macros before putting them in a macro library. Use with /CREATE, /INSERT, and /REPLACE.

### **/TEXT**

See /HELP.

### **/WIDTH=number-of-characters**

Specifies the screen width for listing global symbol names with /NAMES. Defaults to the width of the listing device (usually 80 for terminals and 132 for printers). The maximum allowable width is 132.

## **LOGOUT**

Terminates a terminal session.

### **QUALIFIERS**

### **/BRIEF (default)**

### **/FULL**

Prints a brief logout message (process name, date, and time) or a full logout message (a brief message plus accounting statistics).

### **/FULL**

See /BRIEF.

### **/HANGUP**

### **/NOHANGUP**

Disconnects the phone line when you log out on a dialup terminal.

## MAIL

See Appendix MAIL.

### **MERGE** input1-file-spec,input2-file-spec,... output-file-spec

Merges ordered files into one ordered output file.

#### PARAMETERS

##### **input-file-spec**

Specification of existing file to be merged. All input files must have the same record format and key description, but may have different file organizations. Wildcard characters are not allowed. The file type defaults to DAT.

##### **output-file-spec**

Specification of the file produced by the merge operation. Wildcard characters are not allowed. The file type defaults to the file type of the last input file.

#### QUALIFIERS

##### **/ALLOCATION=file-size**

Qualifies output-file-spec.

Required only to override relative and indexed-sequential input file characteristics. *File-size* is the number of 512-byte blocks to be allocated for the file and must be an integer in the range 1 through 4294967295.

##### **/BUCKET\_SIZE=bucket-size**

Qualifies output-file-spec.

Required only to override input file characteristics. For relative and indexed files, specifies the bucket size in blocks. For input and output files of the same organization, the default is the same as the bucket size of the first input file; otherwise, the default is 1. Bucket size must be an integer in the range 1 through 32.

##### **/CHECK\_SEQUENCE (default)**

##### **/NOCHECK\_SEQUENCE**

Verifies the order of input records.

##### **/COLLATING\_SEQUENCE=sequence**

Names the collating sequence for character data. *Sequence* can be ASCII (default), EBCDIC, or MULTINATIONAL. Note that when you specify EBCDIC, the characters remain in ASCII representation; only the order is changed. The /MULTINATIONAL qualifier specifies the collating sequence of the Multinational character set.

**DCL-80     DCL Commands**  
**MERGE**

**/CONTIGUOUS**

Qualifies output-file-spec.

Required only to override the first input file's characteristics. Specifies that the allocation of blocks for the output file be contiguous. /ALLOCATION must also be specified.

**/DUPLICATES (default)**

**/NODUPLICATES**

Deletes records with duplicate keys from the merge operation. The /NODUPLICATES qualifier is incompatible with the /STABLE qualifier.

**/FORMAT=(option,...)**

Qualifies input-file-spec and output-file-spec.

Specifies record format and size. Possible options for the input file are:

RECORD_SIZE=n	An integer in the range 1 through 32767
FILE_SIZE=n	An integer in the range 1 through 4294967295

Possible options for the output file are:

FIXED=n	A fixed-size record whose maximum size is an integer in the range 1 through 32767 for sequential files; 1 through 16383 for relative files; and 1 through 16383 for indexed sequential files
VARIABLE=n	A variable-size record whose maximum size is an integer in the range 1 through 32767 for sequential files; 1 through 16383 for relative files; 1 through 16383 for indexed sequential files
CONTROLLED=n	A controlled record whose size is an integer in the range 1 through 32767 for sequential files; 1 through 16383 for relative files; and 1 through 16383 for indexed sequential files
SIZE=n	An integer in the range 1 through 255

If /FORMAT is not specified for the output file, the format is based on the merge process selected: if RECORD or TAG merge is selected, the default is the format of the first input file; if ADDRESS or INDEX merge is selected, the default is FIXED.

**/INDEXED\_SEQUENTIAL**

**/RELATIVE**

**/SEQUENTIAL**

Qualifies output-file-spec.

Specifies the organization of the file. For a record or tag merge, the output file format defaults to the organization of the input file. For an indexed merge, the output file must exist and be empty and you must specify the /OVERLAY qualifier.

**/KEY=(option,...)**

Defines a merge key. You can specify /KEY up to 255 times to define 255 different key fields on which to merge. The default is a character data key, beginning in position 1 of the input record for the length of the LRL (longest record length) for the input files, up to a maximum length of 32767 bytes. The following keywords specify position, size, and data type of the key field within the record.

Option	Description
POSITION=start of key	Starting byte of the key within the record, where the first byte of the record is position 1. The value must be an integer in the range 1-32767. The position option is required.
CHARACTER (default)	Data type of the key.
BINARY	
F_FLOATING	
D_FLOATING	
G_FLOATING	
H_FLOATING	
ZONED	
DECIMAL	
PACKED_DECIMAL	
SIZE=n	Size of the key as follows depending on data type: <div><div>CHARACTER—Number of characters specified as an integer in the range 1-32767 (default = 32767)</div><div>BINARY—Then integer value 1 (byte), 2 (word), 4 (longword), 8 (quadword), or 16</div><div>DECIMAL—Number of digits, not counting the sign, specified as an integer in the range 1 to 31</div><div>PACKED_DECIMAL—same as DECIMAL</div></div> <div>Do not specify a size for floating-point data types, whose sizes are fixed at 4 (F_FLOATING), 8 (D_ and G_FLOATING), and 16 (H_FLOATING) bytes. The total size of all keys must not exceed 32767 bytes.</div>

## DCL-82    DCL Commands

### MERGE

Option	Description
NUMBER=key order	Priority of the key specified as an integer in the range 1-255, where 1 means the primary key. The default is the order in which the keys are specified.
ASCENDING(default) DESCENDING	Order in which records are sorted for the key.
SIGNED (default) UNSIGNED	Whether or not a sign is stored (binary keys only)
TRAILING_SIGN (default) LEADING_SIGN	Byte in which sign is stored — first or last (decimal keys only).
OVERPUNCHED_SIGN(default) SEPARATE_SIGN	Whether the sign is superimposed on the decimal value or is separated from the decimal (decimal keys only).

#### **/OVERLAY**

Qualifies output-file-spec.

Writes the output to an existing file which must be empty. By default, a new output file is created for sequential and relative files. If the output file is INDEXED\_SEQUENTIAL, /OVERLAY must be specified.

#### **/RELATIVE**

See /INDEXED\_SEQUENTIAL.

#### **/SEQUENTIAL**

See /INDEXED\_SEQUENTIAL.

#### **/SPECIFICATION=file-spec**

Identifies the specification file to be used in the MERGE operation. Any qualifiers specified in the MERGE command line override the qualifiers in the specification file. The specification file can contain the following qualifiers:

#### **QUALIFIERS**

/CDD\_PATH\_NAME="cdd-path-name"

Specifies a record definition ("cdd-path-name") from the Common Data Dictionary (CDD) if your system has VAX-11 CDD installed. Once the fields have been identified, they may be used in later specification file qualifiers. (The /CDD\_PATH\_NAME qualifier may be used with or in place of the /FIELD qualifier.)

/CHECK\_SEQUENCE (default)  
/NOCHECK\_SEQUENCE

Specifies whether or not MERGE checks the sequence of records in input files.



## MERGE

/COLLATING\_SEQUENCE=(SEQUENCE=sequence[,keyword=,...])

Specifies the collating sequence for character key fields; the collating sequence can be ASCII (the default), EBCDIC, MULTINATIONAL, or a user-defined collating sequence that is specified as a string of characters (single or double) or a range of single characters. Each character and range must be separated by commas and enclosed in parentheses. You can also specify characters by their corresponding octal, decimal, or hexadecimal values, using the radix operators: %O, %D, %X. Specify the quotation mark by doubling its occurrence within quotation marks (""") or by using a radix operator. Specify the null character with a radix operator (such as %X0). You must include in the sequence all characters that appear in the character keys or the character will be ignored (unless the MODIFICATION or FOLD keyword is specified).

Other optional keywords are as follows. Note that the FOLD, MODIFICATION, and IGNORE keywords are processed in the order in which they are specified.

## MODIFICATION

Specifies the change you want to make to the collating sequence (ASCII, EBCDIC, MULTINATIONAL, or user-defined). Use the format MODIFICATION=(character operator character). Specify *character* as it is in the collating sequence and *operator* as >, <, or =. You can specify the following changes to a collating sequence: (1) Equate a character (single or double) to a character (single or double) that has already been assigned a collating value ("a"="A" or "CH"="SH" or "C"="CH"). (2) Collate a character (single or double) after a single character that has already been assigned a collating value ("CH">"C"). (3) Collate a character (single or double) before a character that has already been assigned a collating value ("CH"<"C").

## IGNORE

Specifies the character or range of characters to be initially ignored in the collating sequence (unless two or more strings have compared as equal and (1) TIE\_BREAK is in effect or (2) the Multinational sequence is being used). Specify in the format IGNORE=character (or IGNORE=character range,...).

## FOLD

Gives all lowercase letters the collating value of their uppercase equivalents (the Multinational sequence does this by default).

## DCL-84 DCL Commands

### MERGE

[NO]TIE\_BREAK

Specifies whether or not numeric values are used to break any ties between characters that have equivalent values. (The Multinational sequence breaks ties in this way by default.)

/CONDITION=(NAME=condition-name,TEST=(field-name operator test[logical-operator,...]))

Defines a conditional test that can be used to change the relative order of a record (with the /KEY or /DATA qualifier) or to alter the contents of certain fields of a record (with the /OMIT or /INCLUDE qualifier). *Condition-name* specifies the name of the condition; once defined, you can use the condition name with the /KEY, /DATA, /OMIT, and /INCLUDE qualifiers. *Field-name* specifies the name of the field (defined by the /FIELD qualifier) being tested; *logical operator* specifies the logical (AND or OR) or relational (EQ, NE, GT, GE, LT, or LE) operator used in the test. *Operator-test* specifies the constant for which you are testing. Specify the constant with the following syntax: %D decimal\_digits, %O octal\_digits, %X hexadecimal\_digits, and "character".

/DATA=field-name

/DATA=(IF condition-name THEN "new-contents" ELSE "new-contents")

Specifies the fields to be directed to the output file and their order. (By default, the output file has the same record format as that of the input file.) Only the specified fields will appear in the output field. *Field-name* specifies the previously defined name of a field in a record; *condition-name* specifies a previously defined condition; and *new-contents* is either a constant or a field name that specifies how the record is to be altered.

/FIELD=(NAME=field-name,POSITION:n,SIZE:n, data-type) [DIGITS:n]

Defines the fields in the input files. (You must specify each field in the records to be merged, including key fields, fields to be compared, and fields to be directed to your output file.) *Field-name* cannot have any embedded blanks, must begin with an alphabetic character, and can be no longer than 31 characters. POSITION specifies the position of the field in the record. SIZE specifies the size of a field, according to data type: character data must not exceed 32,767 characters; binary data must be 1, 2, 4, 8, or 16 bytes; and floating-point data has no specified size. *Data-type* specifies the data type of the field; the default is character. Specify *data-type* as:

- CHARACTER (default)
- BINARY[,SIGNED]
- BINARY,UNSIGNED
- D\_FLOATING

- F\_FLOATING
- G\_FLOATING
- H\_FLOATING
- ZONED
- DECIMAL[,SIGNED,TRAILING\_SIGN,OVERPUNCHED\_SIGN]
- DECIMAL,LEADING\_SIGN,SEPARATE\_SIGN[,SIGNED]
- DECIMAL,LEADING\_SIGN,[OVERPUNCHED\_SIGN,SIGNED]
- DECIMAL,[TRAILING\_SIGN],SEPARATE\_SIGN[,SIGNED]
- DECIMAL,UNSIGNED
- PACKED\_DECIMAL

DIGITS specifies the size of a field containing decimal data; *n* cannot exceed 31 digits.

`/INCLUDE=(CONDITION=condition-name,[KEY=...],DATA=...)`

Specifies that records are to be conditionally included (according to a previously defined condition). If you specify multiple `/INCLUDE` qualifiers, the order in which you specify them determines the order in which the input records are tested for inclusion. You unconditionally include any records not previously omitted or included by specifying `/INCLUDE` without a condition. The order of the key fields you specify determines how the internal key is built for merging; the order of the DATA fields determines the way in which the output record is formatted. If you specify a key or data field with `/INCLUDE`, you must define all other key or data fields in the record.

`/KEY=(field-name[,order])`

`/KEY=(IF condition-name THEN value ELSE value)`

Specifies key fields (up to 255) and the order of their priority (unnecessary if you are merging on the entire record using character data). *Field-name* is the name of the field specified in the `/FIELD` qualifier. *Order* can be ASCENDING or DESCENDING. The conditional form of the `/KEY` qualifier specifies a relative order of records; *value* can be a constant or a field name that has been defined in a `/FIELD` qualifier.

`/OMIT=(CONDITION=condition-name)`

Specifies records to be conditionally omitted from the output file (by a previously defined condition). If you specify multiple `/OMIT` qualifiers, the order in which you specify them determines the order in which the input records are tested for omission. You can unconditionally omit any records not previously omitted or included by specifying `/OMIT` without a condition.

## DCL-86     **DCL Commands**

### **MERGE**

**/PAD=**single-character

Specifies a character to be used to fill an incomplete record when you are reformatting a record or comparing strings of unequal length; the null character is the default pad character. The pad character can be a character or a digit (either decimal, octal, or hexadecimal). Enclose characters in quotation marks ("" ) and precede digits with a radix (%X23).

**/STABLE** (default)

**/NOSTABLE** (default)

Arranges records with equal keys in the output file in the same order of the input files (by default, the order is unpredictable).

**/WORK\_FILES=**(disk,...)

Assigns work files to the specified disk and/or diskette. (By default, work files are located in the directory SYS\$SCRATCH; placing them on separate disks with **/WORK\_FILES** permits overlap of MERGE's read/write cycle.)

**/STABLE**

**/NOSTABLE** (default)

Maintains the order of records with identical keys; otherwise, the order is unpredictable. The **/STABLE** qualifier is incompatible with the **/NODUPPLICATES** qualifier.

**/STATISTICS**

**/NOSTATISTICS** (default)

Displays a statistical summary at the end of the merge.

## **MOUNT** device-name,... [volume-label,...] [logical-name]

Makes a volume available for processing. The volume must be physically loaded on a ready device. Volumes mounted privately from a subprocess become owned by the master process.

PARAMETERS

**device-name**

Name of the device (physical name or logical name) on which the volume is physically loaded.

**volume-label**

Label placed on the volume at initialization time. You must specify the label unless the volume is mounted with the **/FOREIGN**, **/NOLABEL**, or **/OVERRIDE=IDENTIFICATION** qualifier. (You must specify the volume-label position in order to specify a logical name.)

**logical-name**

A character string of 1 through 255 characters. If the string contains blanks, enclose it in quotation marks (""). The logical name defaults to DISK\$volume-label for disk volumes. (Do not use a logical name that matches the file name of an executable image in SYS\$SYSTEM.)

**QUALIFIERS**

**/ACCESSED=number-of-directories**

Requires OPER privilege.

Specifies the approximate number of directories that will be in use concurrently on the volume. The value specified with the /ACCESSED qualifier (which can be from 0 through 255) overrides the default set when the volume was initialized.

**/ASSIST (default)**

**/NOASSIST**

Allows operator or user intervention if the MOUNT request fails. Operator replies are written to SYS\$OUTPUT.

**NOTE:** Operator-assisted MOUNT commands will not work unless the OPCOM process is running. SYS\$MANAGER:SYLOGIN.COM creates the symbol MOUNT/NOASSIST for the MOUNT command. If you choose to allow operator-assisted MOUNT commands, you should remove this symbol and start OPCOM by deleting the comment character (!) from the line @SYS\$SYSTEM:STARTUP OPCOM in SYS\$MANAGER:SYSTARTUP.COM.

**/AUTOMATIC (default)**

**/NOAUTOMATIC**

Determines whether MOUNT enables or disables automatic volume switching and labeling for magnetic tape. If you have multiple magnetic tape drives allocated to a volume set, the MTAACP performs the volume switch by sequentially selecting the next available drive allocated to the volume set. The MTAACP expects the next reel of the volume set to be loaded on that drive.

**/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. (Specify the root volume label first, if it is not online.) The *volume-set-name* can be from 1 through 12 alphanumeric characters.

**/BLOCKSIZE=n**

Specifies, for magnetic tape volumes, the default block size. Valid values are in the range 18 through 65,534 for MicroVMS RMS operations and 14 through 65,534 for MicroVMS non-RMS operations. By default, records are written to magnetic tape volumes in 2048-byte blocks. For foreign or unlabeled magnetic tapes, the default is 512 bytes.



## DCL-88    DCL Commands

### MOUNT

You must specify /BLOCKSIZE when you are mounting either tapes that do not have HDR2 labels, or tapes that contain blocks whose size exceeds the default block size (2048 bytes).

#### **/CACHE=(keyword,...)**

#### **/NOCACHE**

For disk devices, overrides the disk caching limits established at system generation time. With the TAPE\_DATA option, enables or disables the write cache for the tape controller specified. Possible keywords are:

[NO]EXTENT[=n]	Enables extent caching to the specified limit. NOEXTENT or EXTENT=0 disables extent caching. Requires OPER privilege.
[NO]FILE_ID[=n]	Enables file identification caching. The value of N must be greater than 1. Requires OPER privilege. NOFILE_ID or FILE_ID=1 disables file identification caching.
LIMIT=n	Specifies the maximum amount of free space in the extent cache in 1/1000's of currently available free space on the disk.
[NO]QUOTA[=n]	Enables quota caching. You normally set N to the maximum number of expected active users for a disk with quotas enabled. NOQUOTA or QUOTA=0 disables quota file caching.
TAPE_DATA	Enables the write cache for a tape device if the tape controller supports a write cache. /NOCACHE is the default for mounting tape devices. If the tape controller does not support a write cache, the option is ignored.
WRITETHROUGH	Disables writeback caching which only writes the file headers of files open for write after the files are closed. Thus, WRITETHROUGH writes file headers 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.

#### **/COMMENT="string"**

Writes the string containing additional information for the operator (when operator assistance is necessary) to the operator log file and the current SYS\$OUTPUT device. The string must contain no more than 78 characters.

#### **/DATA\_CHECK=(keyword,...)**

Checks all read and/or write operations on the disk. Overrides the initialization value. Specify one or both keywords:



READ            Checks all read operations  
WRITE          Checks all write operations (default)

**/DENSITY=*n***

The /DENSITY qualifier is not applicable to the TK50 tape device.

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 the density is supported by the magnetic tape drive. If you do not specify a density for a tape that was previously written, the density defaults to that of the first record on the volume.

**/EXTENSION=*n***

Sets the extend quantity default for all files on the volume. The value *n* can range from 0 through 65535.

**/FOREIGN**

Requires ownership of the volume or VOLPRO privilege for Files-11 volumes.

Indicates the volume is not to be processed by the file system.

**/GROUP**

Requires GRPNAM privilege.

Makes the volume available to all processes with your group number, and makes its name a group logical name.

**/HDR3 (default)**

**/NOHDR3**

Controls whether MicroVMS file header labels are written on magnetic tapes.

By default, MicroVMS file header labels are written on magnetic tape. You can specify /NOHDR3 to write tapes that will be used on other systems that do not process MicroVMS file header labels correctly.

**/INITIALIZE=CONTINUATION**

Specifies that any volume added to the magnetic tape volume set is initialized before you can write to the volume.

**/LABEL (default)**

**/NOLABEL**

Indicates, for magnetic tape volumes, whether the tape contains MicroVMS ANSI labels. Note that /NOLABEL is equivalent to /FOREIGN.

**/MESSAGE (default)**

**/NOMESSAGE**

Writes MOUNT request messages to SYS\$OUTPUT.

**DCL-90     DCL Commands**  
**MOUNT**

**/MOUNT\_VERIFICATION (default)**  
**/NOMOUNT\_VERIFICATION**

Enables mount verification.

**/OVERRIDE=(option,...)**

Requires OPER or VOLPRO privileges to specify /OVERRIDE=(ACCESSIBILITY, EXPIRATION) along with the /FOREIGN qualifier; otherwise, the tape will not be read.

Allows you to override one or more protection checks that the MOUNT command performs. The options are as follows:

**ACCESSIBILITY**                      (For magnetic tapes only.) If the installation allows, this option overrides any character in the Accessibility Field of the volume. The necessity of this option is defined by the installation. That is, each installation has the option of specifying a routine that the magnetic tape file system will use to process this field. By default, MicroVMS provides a routine that checks this field in the following manner. If the magnetic tape was created on a version of MicroVMS that conforms to Version 3 of ANSI, then this option must be used to override any character other than an ASCII space. If a MicroVMS protection is specified and the magnetic tape conforms to an ANSI standard that is later than Version 3, then this option must be used to override any character other than an ASCII 1. To use the ACCESSIBILITY option, you must have the user privilege VOLPRO or be the owner of the volume.

**EXPIRATION**                        (For magnetic tapes only). Allows you to write to a tape that has not yet reached its expiration date. You must have the user privilege (VOLPRO) to override volume protection or your UIC must match the UIC written on 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).

LOCK	Directs MOUNT not to write-lock the volume as a consequence of certain errors encountered while mounting it. Use this option when you are mounting a damaged volume to be repaired using the Verify Utility. (BYPASS privilege will be necessary to actually perform the repair operation.) VOLPRO privilege or ownership of the volume is required to use this option.
OWNER_IDENTIFIER	Allows you to override the processing of the Owner Identifier field of the volume label.
SETID	(For tapes that do not conform to ANSI standards). Allows you to inhibit checks of the file set identifier when you switch reels in a multivolume tape set.

If you specify more than one option, separate them with commas and enclose the list in parentheses.

**/OWNER\_UIC=[uic]**

Requires ownership of the volume or VOLPRO privilege to specify a UIC other than your own.

Specifies an owner UIC for the volume while it is mounted. The default is the UIC initialized for the volume.

**/PROCESSOR=keyword**

Requires OPER privilege.

For magnetic tapes and Files-11 Structure Level 1 disk, requests that the MOUNT command associate an ACP to process the volume. The /PROCESSOR qualifier causes MOUNT to override the default manner in which ACP's are associated with devices.

For Files-11 Structure Level 2 disk, controls block cache allocation.

Possible keywords are:

UNIQUE	For magnetic tape and Files-11 Structure Level 1 disk, creates a new process to execute a copy of the default ACP image for the specified device type or controller.  For Files-11 Structure Level 2 disk, allocates a separate block cache.
SAME:device	For magnetic tape and Files-11 Structure Level 1 disk, uses the same ACP process currently being used by the device specified.  For Files-11 Structure Level 2 disk, takes the block cache allocation from the specified device.

## DCL-92    DCL Commands

### MOUNT

file-spec

Creates 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.

Also, node and directory names are not allowed in the file specification.

This option requires CMKRNL and OPER privilege.

#### **/PROTECTION=(ownership[:access],...)**

Requires VOLPRO privilege for a volume other than your own.

Applies the specified protection to the volume while it is mounted. The default is the protection initialized for the volume. The ownership categories are SYSTEM, OWNER, GROUP, WORLD; the access categories are R (read), W (write), E (create), and D (delete).

#### **/QUOTA (default)**

##### **/NOQUOTA**

Requires VOLPRO privilege for a volume other than your own.

Enforces disk quotas for each user on the volume if the volume contains a quota file.

#### **/REBUILD (default)**

##### **/NOREBUILD**

Controls whether or not MOUNT performs a rebuild operation on a disk volume. If a disk volume has been improperly dismounted (such as during a system failure), it must be rebuilt in order to recover any caching limits that were enabled on the volume at the time of the dismount. By default, MOUNT attempts the rebuild.

The rebuild may consume a considerable amount of time, depending on the number of files on the volume and, if quotas are in use, on the number of different file owners. If you use the /NOREBUILD qualifier, devices can be returned to active use immediately. You can then perform the rebuild later with the DCL command SET VOLUME/REBUILD.

#### **/RECORDSIZE=n**

Specifies, for magnetic 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 18 through 65,534 bytes if you are using MicroVMS RMS, or 14 through 65,534 bytes if you are not using MicroVMS RMS.

**/SHARE**

**/NOSHARE (default)**

Makes the volume available to all users. You can **not** use /SHARE with tape devices.

**/SYSTEM**

Requires SYSNAM privilege.

Makes the volume available to all users of the system and makes its logical name a system logical name.

**/UNLOAD (default)**

**/NOUNLOAD**

Controls whether or not the disk specified in the MOUNT command is unloaded when it is dismounted.

**/WINDOWS=number-of-pointers**

Specifies the number of mapping pointers (used to access data in the file) to be allocated for file windows. The value of *n* can be from 7 through 80; the default is 7. Overrides the initialization value.

**/WRITE (default)**

**/NOWRITE**

Permits WRITE access to the volume.

## **ON event THEN command**

Performs a specified action when an error equal to or greater than the specified error severity level or a CTRL/Y interrupt occurs. Use only in command procedures.

### **PARAMETERS**

#### **event**

One of the following events:

WARNING	Return status of warning (\$SEVERITY equals 0) occurs
ERROR	Return status of error (\$SEVERITY equals 2) occurs
SEVERE_ERROR	Return status of error (\$SEVERITY equals 4) occurs
CONTROL_Y	CTRL/Y character occurs on SYS\$INPUT

#### **command**

Any valid DCL command.

## **OPEN** logical-name file-spec

Opens a file for reading or writing and associates a logical name with the file.

### PARAMETERS

#### **logical-name**

A character string of 1 through 63 characters.

#### **file-spec**

File specification of the file being opened. The file type defaults to DAT. Wildcard characters are not allowed.

### QUALIFIERS

#### **/APPEND**

Opens an existing file for output. New records are added to the end of the file. Incompatible with the /WRITE qualifier.

#### **/ERROR=label**

Transfers control to the location specified by *label* (in a command procedure) if the OPEN operation results in an error. This qualifier overrides any ON condition action specified.

#### **/READ (default)**

Opens the file for reading.

#### **/SHARE[=option]**

Permits other users READ or READ/WRITE access to the file. Specify option as READ or WRITE (default).

#### **/WRITE**

Opens the file for writing. If the file already exists when you open it for WRITE access, a new version is created.

## **PRINT** file-spec,...

Queues one or more files for printing.

### PARAMETERS

#### **file-spec**

Specification of the file to be printed. Wildcard characters are allowed. The plus sign can be used in place of the comma between file specifications. The file type of the first file specified defaults to LIS. The file must not reside on an allocated device. Node names are allowed only when the /REMOTE qualifier is used.



## **QUALIFIERS**

### **/AFTER=time**

#### **/NOAFTER**

Holds the job until the specified time. The time can be specified as an absolute time or a combination of absolute and delta times. If the specified time has passed, the job is queued for printing immediately.

### **/BACKUP**

#### **/NOBACKUP**

Selects files according to the dates of their most recent backups. Relevant only when used with the /BEFORE or /SINCE qualifier.

### **/BEFORE[=time]**

#### **/NOBEFORE**

Selects for printing only those files that are dated before the specified time. You can specify time as an absolute time, as a combination of absolute and delta times, or as one of the following keywords: TODAY (default), TOMORROW, or YESTERDAY. Specified with /BACKUP, /CREATED (default), /EXPIRED, or /MODIFIED.

### **/BURST[=keyword]**

#### **/NOBURST**

Positional qualifier.

Prints a burst page (a flag page printed over the perforation between pages for easy identification of individual files) according to one of the following keywords. If the /BURST qualifier is specified between the PRINT command and the file specifications, it can take either of two keywords:

ALL               Prints a burst page before each file in the job

ONE               Prints a burst page before the first file in the job

The default is the queue specification established in the INITIALIZE/QUEUE command.

To have the /BURST qualifier apply to individual files in a multi-file job, place the qualifier directly after each file that you want to have a burst page. The default is /NOBURST.

### **/BY\_OWNER[=uic]**

#### **/NOBY\_OWNER**

Selects one or more files only if their owner user identification code (UIC) matches the specified UIC. If the BY\_OWNER qualifier is specified without a UIC, the UIC of the current process is assumed.

**DCL-96    DCL Commands**  
**PRINT**

**/CHARACTERISTICS=(characteristic,...)**

Specifies one or more characteristics for printing the job. Use the SHOW QUEUE /CHARACTERISTICS command to display the available characteristics (defined with the DEFINE/CHARACTERISTIC command).

**/CONFIRM**

**/NOCONFIRM (default)**

Issues a request for confirmation before each printing. The following requests are valid:

YES	Print the file
NO	Do not print the file
TRUE	Print the file
FALSE	Do not print the file
1	Print the file
0	Do not print the file
RETURN	Do not print the file
ALL	Continue execution of the command with no further confirmation prompts
CTRL/Z	Stop execution of the command
QUIT	Stop execution of the command

**/COPIES=n**

Positional qualifier.

Specifies the number of copies to print. The value of *n* can be from 1 to 255 and defaults to 1.

If you place the /COPIES qualifier after the PRINT command name, each file in the parameter list is printed the specified number of times. If you specify /COPIES following a file specification, only that file is printed the specified number of times.

**/CREATED (default)**

**/NOCREATED**

Selects files based on their dates of creation. Relevant only when used with the /BEFORE or /SINCE qualifier.

**/DELETE**

**/NODELETE (default)**

Positional qualifier.

Controls whether files are deleted after printing. If you place the /DELETE qualifier after the PRINT command name, all specified files are deleted. If you specify /DELETE after a file specification, only that file is deleted after it is printed.

**/DEVICE=queue-name[:]**

Places the print job in the specified queue (rather than the default queue SYS\$PRINT). This qualifier is synonymous with /QUEUE, except that the /DEVICE qualifier is reserved for special use by DIGITAL. Its usage, therefore, is not recommended.

**/EXCLUDE=(file-spec,...)**

**/NOEXCLUDE**

Excludes any files that match the listed file specifications from the PRINT operation. Wildcard characters are supported for file specifications. However, you cannot use relative version numbers to exclude a specific version.

**/EXPIRED**

**/NOEXPIRED**

Selects files according to the dates on which they will expire. Relevant only when used with the /BEFORE or /SINCE qualifier.

**/FEED (default)**

**/NOFEED**

Positional qualifier.

Automatically inserts form feeds when pages are within 4 lines of the end of the page (line 62 on 66-line forms). You can reset the number of lines per form with the /FORM qualifier. This qualifier does not affect user-formatted files.

**/FLAG[=keyword]**

**/NOFLAG**

Positional qualifier.

Controls whether a flag page is printed preceding a file. The flag page contains the name of the user submitting the job, the job entry number, and other information about the file being printed.

If the /FLAG qualifier is positioned between the PRINT command and the file specifications, it can take either of two keywords:

ALL	Prints a flag page before each file in the job
ONE	Prints a flag page before the first file in the job

To have the /FLAG qualifier apply to individual files in a multi-file job, place the qualifier directly after each file that you want to have a flag page.

**/FORM=type**

Specifies a form type for print queues (defined with the DEFINE/FORM command). Type SHOW QUEUE/FORM to display the available print forms. The default is /FORM=0.

**DCL-98    DCL Commands**  
**PRINT**

**/HEADER**

**/NOHEADER (default)**

Qualifies file-spec.

Prints the name of the file at the top of each page.

**/HOLD**

**/NOHOLD (default)**

Holds the job (until released by a SET QUEUE/ENTRY command).

**/IDENTIFY (default)**

**/NOIDENTIFY**

Displays the queue name and job number of the job when it is queued.

**/JOB\_COUNT=n**

Prints the job *n* times. The value of *n* can be from 1 through 255 and defaults to 1.

**/LOWERCASE**

**/NOLOWERCASE (default)**

Prints the job only on a printer that supports lowercase characters.

**/MODIFIED**

**/NOMODIFIED**

Selects files according to the dates on which they were last modified. Relevant only with the /BEFORE or /SINCE qualifier.

**/NAME=job-name**

Names the job. The name consists of 1 through 39 alphanumeric characters. The default is the name of the first file in the job.

**/NOTE=string**

Specifies a message string of up to 255 characters to appear on the flag page of the job.

**/NOTIFY**

**/NONOTIFY (default)**

Broadcasts a message to your terminal when the job is printed.

**/OPERATOR=string**

Specifies a message of up to 255 characters to be sent to the operator when the job begins to print.

**/PAGES=([lower,]upper)**

Positional qualifier.

Specifies the number of pages to print for each file in the job, or for the specified file. The lower parameter specifies the first page to print; the default is the first page of the file. The upper parameter specifies the last page to print; the default is the last page, but you must include double quotation marks (""") if you do not specify the upper parameter.

**/PARAMETERS=(parameter,...)**

Specifies from one to eight optional parameters to be passed to the job; each parameter can contain up to 255 characters. Enclose parameters containing any special characters or delimiters with quotation marks (""").

**/PASSALL**

**/NOPASSALL (default)**

Positional qualifier.

Specifies whether the symbiont bypasses all formatting and sends the output QIO to the driver with format suppressed for each file in the job, or for the specified file. All qualifiers affecting formatting, as well as the /HEADER, /PAGES and /PAGE\_SETUP qualifiers, will be ignored.

**/PRIORITY=n**

Requires OPER or ALTPRI privilege to raise the priority above the SYSGEN parameter MAXQUEPRI.

Specifies the job's priority. The value of *n* can be from 0 through 255, where 0 is the lowest priority and 255 the highest. The default value of *n* is the value of the SYSGEN parameter DEFQUEPRI.

**/QUEUE=queue-name**

Queues the job to the specified print queue. If this qualifier is omitted, the default is SYS\$PRINT. This qualifier is synonymous with /DEVICE.

**/REMOTE**

Queues the job to SYS\$PRINT on the remote node specified in the file specification (the file must exist on the remote node). You cannot specify any other qualifiers with /REMOTE.

**/RESTART (default)**

**/NORESTART**

Restarts the job after a crash or a STOP/REQUEUE command.

**/SETUP=module,...**

Extracts the specified module from the device control library (containing escape sequence modules for programmable printers) and copies the module to the printer before a file is printed.

## **DCL-100    DCL Commands**

### **PRINT**

#### **/SINCE[=time]**

##### **/NOSINCE**

Selects for printing only those files dated after the specified time. You can specify time as an absolute time, as a combination of absolute and delta times, or as one of the following keywords: TODAY (default), TOMORROW, or YESTERDAY. Specified with /BACKUP, /CREATED (default), /EXPIRED, or /MODIFIED.

#### **/SPACE**

##### **/NOSPACE (default)**

Positional qualifier.

Double spaces all files in the print job, or the specified file (the default is single spacing).

#### **/TRAILER[=keyword]**

##### **/NOTRAILER**

Positional qualifier.

Controls whether a trailer page is printed at the end of a file. The trailer page displays the job entry number as well as information about the user submitting the job and the files being printed.

If the /TRAILER qualifier is positioned between the PRINT command and the file specifications, it can take either of two keywords:

ALL	Prints a trailer page after each file in the job
ONE	Prints a trailer page after the last file in the job

To have the /TRAILER qualifier apply to individual files in a multi-file job, place the qualifier directly after each file that you want to have a trailer page.

#### **/USER=username**

Specifies a user name other than your own as the submitter of the print job.

## **PURGE [file-spec,...]**

Deletes all but the highest numbered version or versions of a file or files.

### **PARAMETERS**

#### **file-spec**

Specification of file to be purged. Wildcard characters are allowed in the directory, file name, and file type fields. No version number can be specified. The default is all files in the current default directory.



**QUALIFIERS**

**/BACKUP**

**/CREATED (default)**

**/EXPIRED**

**/MODIFIED**

Selects files for the purge operation according to the dates of their most recent backups, their creation dates, their expiration dates, or the dates of their last modifications. Relevant only with the /BEFORE and /SINCE qualifiers.

**/BEFORE[=time]**

Purges only those files dated before the specified time. You can specify time as an absolute time, as a combination of absolute and delta times, or as one of the following keywords: TODAY (default), TOMORROW, or YESTERDAY. Specified with /BACKUP, /CREATED (default), /EXPIRED or /MODIFIED.

**/BY\_OWNER[=uic]**

Purges only those files with the specified user identification code. The default UIC is that of the current process.

**/CONFIRM**

**/NOCONFIRM (default)**

Issues a request for confirmation before each purge. The following responses are valid:

YES	Purge
NO	Do not purge
TRUE	Purge
FALSE	Do not purge
1	Purge
0	Do not purge
RETURN	Do not purge
ALL	Continue execution of the command with no further confirmation prompts
CTRL/Z	Stop execution of the command
QUIT	Stop execution of the command

**/CREATED (default)**

See /BACKUP.

**/ERASE**

**/NOERASE (default)**

Erases the specified files from the disk so that the purged data no longer exists physically on the deallocated disk blocks.

## **DCL-102    DCL Commands**

### **PURGE**

#### **/EXCLUDE=(file-spec,...)**

Excludes files from the purge. The qualifier value *file-spec* cannot include a device name. Wildcard characters are supported for file specifications. However, you cannot use relative version numbers to exclude a specific version.

#### **/EXPIRED**

See /BACKUP.

#### **/KEEP=number-of-versions**

Retains the specified number of versions (starting with the highest) of each file. The default is 1.

#### **/LOG**

#### **/NOLOG (default)**

Displays the file specifications of the files as they are purged.

#### **/MODIFIED**

See /BACKUP.

#### **/SINCE[=time]**

Selects for purging only those files dated after the specified time. You can specify time as an absolute time, as a combination of absolute and delta times, or as one of the following keywords: TODAY (default), TOMORROW, or YESTERDAY. Specified with /BACKUP, /CREATED (default), /EXPIRED or /MODIFIED.

## **READ logical-name symbol-name**

Reads a record from SYS\$INPUT, SYS\$COMMAND, or a file opened with the DCL OPEN command and assigns its contents to a symbol.

### **PARAMETERS**

#### **logical-name**

The logical name associated with the input file by the OPEN command, or the logical names SYS\$INPUT or SYS\$COMMAND.

#### **symbol-name**

Name of a symbol to be equated to the contents of the record. The name must be 1 through 255 alphanumeric characters. The symbol becomes a local character symbol. If the symbol has already been defined, the READ command redefines it to the contents of the record.

## QUALIFIERS

### **/DELETE**

Deletes a record from an ISAM file after it has been read. Requires that the ISAM file be opened with the /READ and /WRITE qualifiers.

### **/END\_OF\_FILE=label**

Transfers control to the location specified by *label* (in a command procedure) when the end of the file is reached. The transfer overrides any /ERROR label or ON condition action specified.

### **/ERROR=label**

Transfers control to the location specified by *label* (in a command procedure) when a read error occurs. Overrides any ON condition action specified. The system symbol \$STATUS retains the error code.

### **/INDEX=key-of-reference**

Specifies the index to be used to look up keys when reading an ISAM file. The default value is 0, the primary index.

### **/KEY=key-value**

Reads a record with the key that matches the specified value. Enclose the value in quotation marks ("). Binary and integer keys are not allowed.

### **/MATCH=option**

Specifies the match algorithm to be used when searching for matching keys.

EQ	Select keys equal to the match value (default)
GE	Select keys greater than or equal to the match value
GT	Select keys greater than the specified key

### **/NOLOCK**

Specifies that the record not be locked and enables a record to be read that has been locked by other accessors. By default, records are locked as they are read and unlocked on the next I/O operation on the file.

### **/PROMPT=prompt**

Specifies an alternate prompt if you are reading from the terminal. Enclose the prompt in quotation marks (") if it contains spaces, special characters, or lower case characters. The default prompt is DATA:.

### **/TIME\_OUT**

### **/NOTIME\_OUT (default)**

Specifies a number of seconds after which READ is terminated if no input is received. If you enter /TIME\_OUT, you must specify a value from 0 through

**DCL-104    DCL Commands**  
**READ**

255. If you enter both the /TIME\_OUT and /ERROR qualifiers, and if the time limit expires, the error branch is taken.

**RECALL [command]**

Displays previously entered commands on the screen for subsequent execution.

**PARAMETERS**

**command**

The first few characters or the number of the command you wish to recall. The specified characters should be sufficient to make the command unique since the most recently issued command line that matches the specified characters is recalled. The number of the command can be from 1 to 20 (where 1 is the last command entered) and defaults to 1.

**QUALIFIERS**

**/ALL**

Displays all the commands (and their numbers) available for recall.

**RENAME input-file-spec,... output-file-spec**

Changes all or part of a file specification.

**PARAMETERS**

**input-file-spec**

Specification of file to be renamed. Wildcard characters are allowed.

**output-file-spec**

New file specification. Wildcard characters are allowed. Parts of the output file specification that are omitted or replaced by wildcard characters default to the corresponding parts of the input file specification, except for the version number. A version number is determined (1) by the output file version number if specified explicitly, (2) by the input file version if a wildcard is used for the input or output file type, or (3) by the next higher version number or a version number of 1 if the version number is not specified.

## RENAME

## QUALIFIERS

**/BACKUP****/CREATED (default)****/EXPIRED****/MODIFIED**

Selects files for the rename operation according to the dates of their most recent backups, their creation dates, their expiration dates, or the dates of their last modifications. Relevant only with the **/BEFORE** and **/SINCE** qualifiers.

**/BEFORE[=time]**

Renames only those files with dates that precede the specified time. You can specify time as an absolute time, as a combination of absolute and delta times, or as one of the following keywords: **TODAY** (default), **TOMORROW**, or **YESTERDAY**. Specified with **/BACKUP**, **/CREATED** (default), **/EXPIRED** or **/MODIFIED**.

**/BY\_OWNER[=uic]**

Renames only those files with the specified user identification code. The default UIC is that of the current process.

**/CONFIRM****/NOCONFIRM (default)**

Issues a request for confirmation before each file is renamed. The following responses are valid:

YES	Rename the file
NO	Do not rename the file
TRUE	Rename the file
FALSE	Do not rename the file
1	Rename the file
0	Do not rename the file
RETURN	Do not rename the file
ALL	Continue execution of the command with no further confirmation prompts
CTRL/Z	Stop execution of the command
QUIT	Stop execution of the command

**/CREATED (default)**

See **/BACKUP**.

**RENAME****/EXCLUDE(=file-spec,...)**

Excludes the specified files from the rename operation. The qualifier value file-spec cannot include a device name. Wildcard characters are supported for file specifications. However, you cannot use relative version numbers to exclude a specific version.

**/EXPIRED**

See /BACKUP.

**/LOG****/NOLOG (default)**

Displays the file specification of each file as it is renamed.

**/MODIFIED**

See /BACKUP.

**/NEW\_VERSION (default)****/NONEW\_VERSION**

Assigns a new version number if an output file specification is the same as that of an existing file. The /NONEW\_VERSION qualifier displays an error message if an output file specification is the same as that of an existing file.

**/SINCE[=time]**

Renames only those files dated after the specified time. You can specify time as an absolute time, as a combination of absolute and delta times, or as one of the following keywords: TODAY (default), TOMORROW, or YESTERDAY. Specified with /BACKUP, /CREATED (default), /EXPIRED or /MODIFIED.

**REPLY "message-text"**

Broadcasts a message to a terminal or terminals.

**PARAMETERS****message-text**

Text of the message. The text must be 1 through 128 characters. Enclose the text in quotation marks (""") if it contains spaces, special characters, or lowercase characters.



## QUALIFIERS

### **/ABORT=identification-number**

Sends a message to the user or magnetic tape file system corresponding to the unique identification number and cancels the request.

### **/ALL**

Requires OPER privilege.

Broadcasts a message to all terminals that are attached to the system, that have broadcast-message reception enabled, and that are turned on. Incompatible with /USERNAME and /TERMINAL.

### **/BELL**

Rings a bell at the terminal receiving a message when issued with the /ALL, /TERMINAL, or /USER qualifiers; two bells when issued with /URGENT; and three bells when issued with /SHUTDOWN.

### **/BLANK\_TAPE=identification-number**

Requires VOLPRO privilege.

Sends a message to the magnetic tape file system indicated by the identification number to override the checking of volume label information. The volume label must be specified in the message text parameter. The current terminal must be enabled as an operator terminal for TAPES.

### **/DISABLE[=(keyword,...)]**

Requires OPER privilege.

Restores to normal (that is, nonoperator) status a terminal at which the command is issued or whose name is specified in the message text parameter. The /DISABLE qualifier cannot be issued from a batch job. To restrict the types of messages displayed on an operator's terminal, specify one of the following keywords:

CENTRAL	Inhibits messages sent to the central system operator
DEVICES	Inhibits messages pertaining to mounting disks
DISKS	Inhibits messages pertaining to mounting and dismounting disk volumes
NETWORK	Inhibits messages pertaining to networks; the keyword CENTRAL must also be specified to inhibit network messages
OPER1 through OPER12	Inhibits messages sent to operators identified as OPER1 through OPER12
PRINTER	Inhibits messages pertaining to print requests

**DCL-108    DCL Commands**  
**REPLY**

SECURITY	Inhibits/allows messages pertaining to security events. Requires SECURITY privilege.
TAPES	Inhibits/allows messages pertaining to mounting and dismounting tape volumes.

**/ENABLE[=(keyword,...)]**

Requires OPER privilege.

Designates as an operator's terminal the terminal at which the REPLY command is issued. Cannot be issued from a batch job.

CENTRAL	Displays messages sent to the central system operator
DEVICES	Displays messages pertaining to mounting disks
DISKS	Displays messages pertaining to mounting and dismounting disk volumes
NETWORK	Displays messages pertaining to networks; the keyword CENTRAL must also be specified to inhibit network messages
OPER1 through OPER12	Displays messages sent to operators identified as OPER1 through OPER12
PRINTER	Displays messages pertaining to print requests
SECURITY	Inhibits/allows messages pertaining to security events. Requires SECURITY privilege.
TAPES	Inhibits/allows messages pertaining to mounting and dismounting tape volumes.

**/INITIALIZE\_TAPE=identification-number**

Sends a message to the magnetic tape file system indicated by the identification number to initialize a magnetic tape volume. This qualifier can be used whenever the file system requests the mounting of a new volume. The system performs normal protection and expiration checks before initializing the volume. The current terminal must be enabled as an operator terminal for TAPES.

**/LOG**

**/NOLOG**

Requires OPER privilege.

Closes the current operator's log file and opens a new one. (The /NOLOG qualifier does not open a new log file.) The current terminal must be enabled as an operator terminal.

**/NOTIFY(default)**

**/NONOTIFY**

Sends a message describing success back to the originating terminal.

**/PENDING=identification-number**

Requires OPER privilege.

Sends a message to the user specified by the identification number and prevents the user from entering other commands until the operator fulfills or aborts the request. The current terminal must be enabled as an operator terminal.

**/SHUTDOWN**

Sends a message beginning "SHUTDOWN..."; if used with /BELL, rings three bells at terminals receiving the message.

**/STATUS**

Requires OPER privilege.

Reports the current operator status and all outstanding user requests for the terminal from which this command was entered. The current terminal must be enabled as an operator terminal.

**/TEMPORARY**

Designates the terminal at which the command is issued to be an operator's terminal for the current interactive session only. This qualifier is meaningful only when used with the /ENABLE qualifier.

**/TERMINAL=(terminal-name,...)**

Requires OPER privilege.

Broadcasts the message to specified terminals, where *terminal-name* is the device name of the terminal. Incompatible with /ALL and /USERNAME.

**/TO=identification-number**

Requires OPER privilege.

Sends a message to the user or file system specified by the identification number and completes the request. The current terminal must be enabled as an operator terminal.

**/URGENT**

Sends a message beginning "URGENT..."; if used with the /BELL qualifier, rings two bells at terminals receiving the message.

**/USERNAME[(username,...)]**

Requires OPER privilege.

Broadcasts a message to all terminals at which users are logged in to the system, or only to the terminals of the specified users. Overrides any NOBROADCAST settings at users' terminals.

**/WAIT**

Sends message synchronously and waits.

## DCL-110    DCL Commands

### REQUEST

#### REQUEST "message-text"

Displays a message at the operator's terminal and optionally requests a reply. All messages are logged at the operator's console and in the operator's log file, if that file is initialized.

**NOTE:** To use this command, you must start the OPCOM process at boot-time by specifying the DCL command @SYS\$SYSTEM:STARTUP OPCOM in the site-specific startup command file, SYS\$MANAGER:SYSTARTUP.COM.

#### PARAMETERS

##### message-text

The text of the message to be displayed. The string can be up to 128 characters. If the string contains spaces, special characters, or lowercase characters, enclose it in quotation marks ("").

#### QUALIFIERS

##### /REPLY

Requests a reply to the message and issues a unique identification number to which the operator sends the response. You receive a message that the operator has been notified; you cannot enter any commands until the operator responds. If you press CTRL/C before the operator responds, you can then enter another message to the operator, or press CTRL/Z to cancel the request.

##### /TO[=(operator,...)]

Specifies one or more operators to whom you wish to send the message. Possible keywords are:

CENTRAL	Sends the message to the central system operator
DEVICES	Sends the message to operators who mount and dismount disks
DISKS	Sends the message to operators who mount and dismount disk volumes
NETWORK	Sends the message to the network operator
OPER1 through OPER12	Sends the message to operators identified as OPER1 through OPER12
PRINTER	Sends the message to operators designated to handle print requests
SECURITY	Sends the message to operators designated to respond to security-related requests
TAPES	Sends the message to operators designated to mount and dismount tape volumes

## **RETURN** [status-code]

Terminates a GOSUB subroutine and returns control to the command following the calling GOSUB command.

### PARAMETER

#### **status-code**

Longword (integer) value giving the exit status of the subroutine. (This value is assigned to the global symbol \$STATUS and the lower three bits determine the value of the global symbol \$SEVERITY.)

## **RUN** file-spec (image)

Executes an image within the context of your process.

### PARAMETERS

#### **file-spec**

Specification of a file containing an executable image. The file type defaults to EXE. Wildcard characters are not allowed.

### QUALIFIERS

#### **/DEBUG**

#### **/NODEBUG**

Executes the image under control of the debugger. The default is /DEBUG if the image is linked with /DEBUG and /NODEBUG if the image is linked without /DEBUG. The /DEBUG qualifier is invalid if the image is linked with /NOTRACEBACK. The /NODEBUG qualifier overrides the effect of LINK /DEBUG.

## **RUN** file-spec (process)

Creates a subprocess or a detached process to run an image and deletes the process when the image completes execution. If you specify any of the qualifiers except /UIC or /DETACHED, the RUN command creates a subprocess. A detached process is created if the /UIC qualifier is specified and you have the DETACH user privilege.

### PARAMETERS

#### **file-spec**

Specification of a file containing an executable image. The file type defaults to EXE. Wildcards are not allowed.

## DCL-112    DCL Commands

### RUN

#### QUALIFIERS

##### **/ACCOUNTING (default)**

##### **/NOACCOUNTING**

Requires ACNT privilege to disable accounting.

Logs accounting records for the created process.

##### **/AST\_LIMIT=quota**

Specifies the maximum number of asynchronous system traps (ASTs) that the created process can have outstanding. The default quota is that established at system generation time. The minimum required for any process to execute is 2. The AST limit quota is nondeductible.

##### **/AUTHORIZE**

##### **/NOAUTHORIZE (default)**

Requires DETACH privilege.

Searches the user authorization file to validate a detached process when the image to be executed is the system login image (LOGINOUT.EXE). The /NOAUTHORIZE qualifier creates a detached process that runs under the control of the command interpreter.

##### **/BUFFER\_LIMIT=quota**

Specifies the maximum amount of memory, in bytes, that the process can use for buffered I/O operations or temporary mailbox creation. The quota default is that established at system generation time. The minimum amount required for any process to execute is 1024 bytes; the buffer limit quota is pooled.

##### **/DELAY=delta-time**

Places the created process in hibernation and awakens it after a specified time interval. If you specify both /DELAY and /INTERVAL, the first wakeup request occurs at the time specified by /DELAY and all subsequent wakeups occur at intervals as specified by /INTERVAL.

##### **/DETACHED**

##### **/NODETACHED**

Creates a detached process with the same user identification code (UIC) as the current process. (Use the /UIC qualifier to create a detached process with a different UIC.) By default, the detached process has the same resource quotas as the current process; the DETACH privilege allows you to specify quotas for the detached process. The maximum number of detached processes that you can create is limited to the quota defined by MAX\_DETACH in your user authorization file, unless you have the DETACH privilege.



**/DUMP**

**/NODUMP (default)**

When an image terminates due to an unhandled error, /DUMP writes the contents of the address space to the file named SYS\$LOGIN:IMAGEDUMP.DMP.

**/ENQUEUE\_LIMIT=quota**

Specifies the maximum number of locks that a process can have outstanding at any one time. The default quota is that established at system generation time. The minimum required for any process to operate is 2.

**/ERROR=file-spec**

Defines an equivalence name string of from 1 to 63 alphanumeric characters for the logical device name SYS\$ERROR. The logical name and equivalence name are placed in the process logical name table for the created process. (This qualifier is ignored if you are running SYS\$SYSTEM:LOGINOUT.)

**/EXTENT=quota**

Specifies the maximum size to which the image being executed in the process can increase its physical memory size. The default quota is that established at system generation time. The minimum value required for any process to execute is 10 pages. The extent quota is nondeductible.

**/FILE\_LIMIT=quota**

Specifies the maximum number of files that a process can have open at any one time. The default quota is that established at system generation time. The minimum amount required for any process to execute is 2. The file limit quota is pooled.

**/INPUT=file-spec**

Defines an equivalence name string of from 1 to 63 characters for SYS\$INPUT. The logical name and equivalence name are placed in the process logical name table for the created process.

**/INTERVAL=delta-time**

Requests that the created process be placed in hibernation and awakened at regularly scheduled intervals. If you specify the /DELAY or /SCHEDULE qualifier with the /INTERVAL qualifier, 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.

**/IO\_BUFFERED=quota**

Specifies the maximum number of system-buffered I/O operations that the created process can have outstanding at any one time. The default quota is that

## **DCL-114    DCL Commands**

### **RUN**

established at system generation time. The minimum required for any process to execute is 2. The buffered I/O quota is nondeductible.

#### **/IO\_DIRECT=quota**

Specifies the maximum number of direct I/O operations that the created process can have outstanding at any one time. The default quota is that established at system generation time. The minimum required for any process to execute is 2. The direct I/O quota is nondeductible.

#### **/JOB\_TABLE\_QUOTA=quota**

Allows you to specify a quota for a detached process's job-wide logical name table.

Note that the /JOB\_TABLE\_QUOTA qualifier is relevant only for detached processes. If the /JOB\_TABLE\_QUOTA is specified in a RUN command which results in the creation of a subprocess, it will be ignored.

#### **/MAILBOX=unit**

Specifies the unit number of a mailbox to receive a termination message when the created process is deleted. (If no mailbox is specified, the creating process receives no notification of the subprocess's deletion.)

#### **/MAXIMUM\_WORKING\_SET=quota**

Specifies the maximum size to which the image being executed in the process can increase its working set size. The default quota is that established at system generation time. The minimum value required for any process to execute is 10 pages. The maximum working set quota is nondeductible.

#### **/OUTPUT=file-spec**

Defines an equivalence name string of from 1 to 63 characters for the logical device name SYS\$OUTPUT. Both the equivalence name and the logical name are placed in the process logical name table for the created process.

#### **/PAGE\_FILE=quota**

Specifies the maximum number of pages that can be allocated in the paging file for the process. The default quota is that established at system generation time. The minimum value required for a process to execute is 256 pages. The paging file quota is pooled.

#### **/PRIORITY=n**

Requires ALTPRI privilege to set the priority higher than your base priority. Specifies the base priority at which the created process will execute. The value of *n* is a decimal number from 0 through 31. The default priority is that of the current process.

**/PRIVILEGES=(keyword,...)**

Requires SETPRV privilege to specify privileges that you do not have.

Defines user privileges for the created process. By default, the created process has the same privileges as its creator. If you specify a version number (or semicolon) in the *file-spec* parameter, the current process privileges are used, overriding any privileges specified with the /PRIVILEGES qualifier. The following table lists process privileges:

[NO]ACNT	Create processes for which no accounting messages are written
[NO]ALL	Have all privileges
[NO]ALLSPOOL	Allocate spooled devices
[NO]ALTPRI	Set priority values
[NO]BUGCHK	Make bug check error log entries
[NO]BYPASS	Bypass UIC protection
[NO]CMEXEC	Change mode to executive
[NO]CMKRNL	Change mode to kernel
[NO]DETACH	Create detached processes
[NO]DIAGNOSE	Issue diagnostic I/O requests
[NO]EXQUOTA	Exceed quotas
[NO]GROUP	Control other processes in the same group
[NO]GRPNAM	Place names in the group logical name table
[NO]GRPPRV	Access group objects
[NO]LOG_IO	Issue logical I/O requests to a device
[NO]MOUNT	Issue a mount volume QIO request
[NO]NETMBX	Create a network device
[NO]OPER	Perform operator functions
[NO]PFNMAP	Create or delete sections mapped by page frame number
[NO]PHY_IO	Issue physical I/O requests to a device
[NO]PRMCB	Create permanent common event flag clusters
[NO]PRMGBL	Create permanent global sections
[NO]PRMJNL	Create a permanent journal
[NO]PRMMBX	Create permanent mailboxes
[NO]PSWAPM	Alter swap mode
[NO]READALL	Bypass existing restrictions on reading a file
[NO]SECURITY	Perform security-related functions
[NO]SETPRV	Give higher privileges to other processes
[NO]SHARE	Assign a channel to a device, even if the channel is allocated to another process

## DCL-116    DCL Commands

### RUN

[NO]SHMEM	Create data structures in shared memory
[NO]SYSGBL	Create system global sections
[NO]SYSLCK	Request locks on systemwide resources
[NO]SYSNAM	Place names in the system logical name table
[NO]SYSPRV	Access system objects
[NO]TMPJNL	Create a temporary journal
[NO]TMPMBX	Create a temporary mailbox
[NO]VOLPRO	Override volume protection
[NO]WORLD	Control all other processes in the system

#### **/PROCESS\_NAME=process-name**

Specifies a name of from 1 to 15 characters for the created process. The process name is implicitly qualified by the group number of the process's UIC. By default, the name is null.

#### **/QUEUE\_LIMIT=quota**

Specifies the maximum number of timer queue entries that the created process can have outstanding at any one time. The default quota is that established at system generation time. The timer queue entry quota is pooled.

#### **/RESOURCE\_WAIT (default)**

#### **/NORESOURCE\_WAIT**

Places the created process in a wait state when a resource required for a particular function is not available. The /NORESOURCE\_WAIT qualifier generates an error status code when a resource is unavailable.

#### **/SCHEDULE=absolute-time**

Places the created process in hibernation and awakens it at the specified time.

#### **/SERVICE\_FAILURE**

#### **/NOSERVICE\_FAILURE (default)**

Signals an exception condition if an error occurs during a system service request. By default, an error status code is returned to the process.

#### **/SUBPROCESS\_LIMIT=quota**

Specifies the maximum number of subprocesses that the created process is allowed to create. The default quota is that established at system generation time. The subprocess limit is pooled.

**/SWAPPING (default)**

**/NOSWAPPING**

Requires PSWAPM privilege to inhibit swapping.

Permits the process to be swapped.

**/TIME\_LIMIT=quota**

Specifies the maximum amount of CPU time (in delta time) to be allocated to the created process. The resolution is to 10 milliseconds. When the time expires, the process is deleted. The default limit is that established at system generation time (usually infinite). A CPU time limit of 0 specifies that CPU time is not restricted. The time limit quota is deductible.

**/UIC=uic**

Specifies that the created process be a detached process and assigns it a UIC.

**/WORKING\_SET=default**

Specifies the number of pages in the working set of the created process. The default working set size is that established at system generation time. The minimum number of pages required for a process to execute is 10 pages. The value specified cannot be greater than the quota specified with /MAXIMUM\_WORKING\_SET. The maximum working set quota is nondeductible.

**RUNOFF**

See Appendix DSR.

**RUNOFF/CONTENTS**

See Appendix DSR.

**RUNOFF/INDEX**

See Appendix DSR.

**SEARCH file-spec,... search-string,...**

Displays all occurrences of the specified string within the specified files.

## **DCL-118    DCL Commands**

### **SEARCH**

#### PARAMETERS

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

Specification of file to be searched. Wildcard characters are allowed.

##### **search-string[,...]**

Character string to be located. Enclose strings containing lowercase letters, blanks or other nonalphanumeric characters in quotation marks.

#### QUALIFIERS

##### **/EXACT**

##### **/NOEXACT (default)**

Distinguishes between uppercase and lowercase characters.

##### **/EXCLUDE=(file-spec,...)**

Excludes the specified files from the search. Do not include device and directory fields in the filespec. Wildcards characters are supported for file specifications. However, you cannot use relative version numbers to exclude a specific version.

##### **/FORMAT=option**

Formats output in one of four ways:

DUMP	Displays both control and nonprintable characters as ANSI mnemonics.
NONULLS	Same as DUMP, but removes any null characters.
PASSALL	Does not translate control and nonprintable characters; the terminal driver does not actually pass 8-bit characters to the terminal unless SET TERMINAL/PASTHRU or SET TERMINAL/EIGHT_BIT are already in effect.
TEXT	Displays control characters as ANSI mnemonics; displays terminal formatting characters as: <HT> , <CR> , <LF> , <VT> , and <FF> .

##### **/HEADING (default)**

##### **/NOHEADING**

Includes file names in the output file and displays 30 asterisks as a window separator between groups of lines that belong to different files. (The /WINDOW qualifier displays 15 asterisks as a separator between windows.)

##### **/LOG**

##### **/NOLOG (default)**

Types a message to SYS\$OUTPUT for each file searched. The message includes the file name, the number of records, and the number of matches for each file searched.



**/MATCH=option**

Interprets multiple search strings in one of the following ways:

AND	A match occurs only if the record contains all the strings.
NOR	A match occurs only if the record contains none of the strings.
NAND	A match occurs only if the record does not contain all of the strings.
OR	A match occurs if the record contains any of the strings. (default).

**/NUMBERS**

**/NONUMBERS (default)**

Includes source line numbers in the output.

**/OUTPUT[=file-spec]**

**/NOOUTPUT**

Specifies an output file. The default is SYS\$OUTPUT.

**/REMAINING**

**/NOREMAINING (default)**

Includes in the output all records from the first matched record to the end of the file. This qualifier overrides *n2* in /WINDOW, but allows /WINDOW=*n1*.

**/STATISTICS**

**/NOSTATISTICS (default)**

Determines whether or not statistics about the search are displayed.

**/WINDOW[=(*n1* [, *n2* ])]**

**/NOWINDOW (default)**

Specifies the number of lines to be displayed with the search string. If you specify the /WINDOW qualifier without *n1* and *n2*, two lines above the search string, the search string, and the two lines below the search string are included in the output. If you specify *n1* and *n2*, the /WINDOW qualifier displays *n1* lines above the search string, the search string, and *n2* lines below the search string. If you specify /WINDOW with a single number (*n1*), *n1* specifies the number of lines to display including the search string: half the lines precede the matched search string, and half follow it. (If *n* is odd, 1 line is added to the lines following the matched search string.) If you specify /WINDOW=0, the file name of each file containing a match (but no records) is included in the output. If you omit the /WINDOW qualifier, only the line containing a match is displayed.

**SET ACCOUNTING**

Enables or disables the logging of various activities in the accounting log file SYS\$MANAGER:ACCOUNTNG.DAT, or closes the current accounting log file and opens a new one with a version number incremented by 1.

## **DCL-120    DCL Commands**

### **SET ACCOUNTING**

#### QUALIFIERS

**/DISABLE[=(keyword,...)]**

**/ENABLE[=(keyword,...)]**

Disables or enables the logging of the specified activities recorded in the accounting log file. If you specify only /DISABLE, the logging of all activities is disabled. If you specify only /ENABLE, the logging of all activities is enabled. The following keywords specify the activities:

BATCH	Batch job termination
DETACHED	Detached job termination
IMAGE	Image activation
INTERACTIVE	Interactive job termination
LOGIN_FAILURE	Login failures
MESSAGE	User messages
NETWORK	Network job termination
PRINT	Print jobs
PROCESS	Process termination
SUBPROCESS	Subprocess termination

**/NEW\_FILE**

Closes the current accounting file and opens a new version of that file.

#### **SET ACL**

See Appendix ACL.

#### **SET AUDIT**

Requires the SECURITY privilege.

Enables security auditing to send alarms to terminals that have been enabled as security operators (see REPLY/ENABLE) whenever the system detects specified events. If you enable security alarms, you must delete the comment delimiter (!) from the following line in SYS\$MANAGER:SYSTARTUP.COM:

**\$! @SYS\$SYSTEM:STARTUP OPCOM**

## QUALIFIERS

**/ALARM**

Sends alarm messages to all terminals enabled as security operators. Both /ALARM and either /DISABLE or /ENABLE are required.

**/DISABLE=(keyword,...)****/ENABLE=(keyword,...)**

Enables or disables security auditing for the specified events. The /ENABLE qualifier requires the /ALARM qualifier; /DISABLE overrides /ENABLE. Specify events to be audited with the following keywords:

ACL	An event requested by an Access Control List (ACL), including ACLs on files and global sections												
ALL	All possible events												
AUDIT	An event resulting from the execution of a SET AUDIT command.												
AUTHORIZATION	The modification of any portion of the system or network user authorization file (UAF), including any password changes; the modification of any portion of the rights database												
BREAKIN=(keyword,...)	The occurrence of one or more of the following classes of breakin attempts, as specified by one or more of the keywords below: <table><tr><td>ALL</td><td>All possible sources of breakins, as defined by the remaining keywords</td></tr><tr><td>DETACHED</td><td>Detached process breakin attempt</td></tr><tr><td>DIALUP</td><td>Dialup breakin attempt</td></tr><tr><td>LOCAL</td><td>Local breakin attempt</td></tr><tr><td>NETWORK</td><td>Network server breakin attempt</td></tr><tr><td>REMOTE</td><td>Remote breakin attempt</td></tr></table>	ALL	All possible sources of breakins, as defined by the remaining keywords	DETACHED	Detached process breakin attempt	DIALUP	Dialup breakin attempt	LOCAL	Local breakin attempt	NETWORK	Network server breakin attempt	REMOTE	Remote breakin attempt
ALL	All possible sources of breakins, as defined by the remaining keywords												
DETACHED	Detached process breakin attempt												
DIALUP	Dialup breakin attempt												
LOCAL	Local breakin attempt												
NETWORK	Network server breakin attempt												
REMOTE	Remote breakin attempt												

**DCL-122    DCL Commands**  
**SET AUDIT**

FILE\_ACCESS=(keyword,...)

The occurrence of file and global section access events (regardless of the value specified in the file's access control list, if any). You can specify one or more of the following keywords to describe the type of file access event.

ALL                    All types of file access events,  
                         as defined by the remaining  
                         keywords

BYPASS  
[:access,...]        Successful file access due to the  
                         use of the BYPASS privilege

FAILURE  
[:access,...]        Unsuccessful file access

GRPPRV  
[:access,...]        Successful file access due to the  
                         use of the GRPPRV privilege

READALL  
[:access,...]        Successful file access due to the  
                         use of the READALL privilege

SUCCESS  
[:access,...]        Successful file access

SYSPRV  
[:access,...]        Successful file access due to the  
                         use of the SYSPRV privilege

Most of the keywords permit you to optionally define the type of file access that was obtained with the following keywords:

ALL                    All types of file access events,  
                         as defined by the remaining  
                         keywords; if no access types are  
                         specified, ALL is assumed by  
                         the system

READ                  Read access

WRITE                Write access

EXECUTE              Execute access

DELETE               Delete access

CONTROL              Owner access

INSTALL

The occurrence of any INSTALL operations

LOGFAILURE=(keyword,...)

The occurrence of one or more of the following classes of login failure, as specified by one or more of the keywords:

ALL	All possible types of login failures, as defined by the remaining keywords
BATCH	Batch process login failure
DETACHED	Detached process login failure
DIALUP	Dialup interactive login failure
LOCAL	Local interactive login failure
NETWORK	Network server task login failure
REMOTE	Interactive login failure from another network node, for example, with a SET HOST command
SUBPROCESS	Subprocess process login failure

LOGIN=(keyword,...)

The occurrence of one or more of the following classes of logins, as specified by one or more of the keywords:

ALL	All possible sources of logins, as defined by the remaining keywords
BATCH	Batch process login
DETACHED	Detached process login
DIALUP	Dialup interactive login
LOCAL	Local interactive login
NETWORK	Network server task login
REMOTE	Interactive login from another network node, for example, with a SET HOST command
SUBPROCESS	Subprocess process login

## DCL-124    DCL Commands

### SET AUDIT

LOGOUT=(keyword,...)

The occurrence of one or more of the following classes of logouts, as specified by one or more of the keywords:

ALL	All possible sources of logouts, as defined by the remaining keywords
BATCH	Batch process logout
DETACHED	Detached process logout
DIALUP	Dialup interactive process logout
LOCAL	Local interactive process logout
NETWORK	Logout by a network server task
PROCESS	Subprocess or detached process logout
REMOTE	Logout of a process that logged in interactively from another network node

MOUNT

The issuing of a MOUNT or DISMOUNT request

### SET BROADCAST =(class-name,...)

Enables you to select the kinds of messages to be broadcast to your terminal.

#### PARAMETERS

##### class-name

The class of message that you want to enable or disable for broadcast to your terminal. Specify the class of message with one or more of the following keywords:

ALL	All messages
[NO]DCL	Messages issued by DCL
[NO]GENERAL	All normal REPLY messages or messages from \$BRDCST
[NO]MAIL	Messages giving notification of mail
NONE	No messages
[NO]OPCOM	Messages issued by OPCOM
[NO]PHONE	Messages from the Phone Utility



[NO]QUEUE	Messages about print or batch jobs issued by the queue manager
[NO]SHUTDOWN	Messages issued from REPLY/ID=SHUTDOWN
[NO]URGENT	Messages issued from REPLY/ID=SHUTDOWN
[NO]USERn	Messages from the specified user groups; <i>n</i> can be from 1 through 16

## **SET [NO]CONTROL [= (T,Y)]**

Enables or disables CTRL/Y, which interrupts a command and returns you to DCL command level, and/or CTRL/T, which momentarily interrupts a command to print a line of statistics. The default keyword is Y.

CTRL/C responds as a CTRL/Y unless the current image has a special action routine defined for CTRL/C.

## **SET DAY**

Requires OPER privilege.

Sets the default day type specified in the user authorization file (UAF) for the current day.

### **QUALIFIERS**

#### **/DEFAULT**

Overrides any previous SET DAY specification and sets the normal UAF defaults as today's day type.

#### **/LOG**

#### **/NOLOG (default)**

Displays the new SET DAY information on the terminal.

#### **/PRIMARY**

Sets today until midnight to a primary day.

#### **/SECONDARY**

Sets today until midnight to a secondary day.

**DCL-126    DCL Commands**  
**SET DEFAULT**

**SET DEFAULT partial-file-spec**

Sets your default device and directory specifications.

**PARAMETERS**

**partial-file-spec**

Device and/or directory name. A device name must be terminated with a colon. A directory name must be enclosed in brackets. A logical name can be used for the specification but it must constitute at least the device part of the specification. The minus sign wildcard can be used to specify the next higher directory from the current default.

**SET DEVICE device-name**

Requires OPER privilege.

Sets various characteristics for a device. You can use the SHOW DEVICE/FULL command to find out the state of these characteristics.

**PARAMETERS**

**device-name**

Name of the affected device.

**QUALIFIERS**

**/AVAILABLE**

**/NOAVAILABLE**

Specifies that the disk is available for mounting.

**/DUAL\_PORT**

**/NODUAL\_PORT**

Enables the port seize logic in the device driver of the specified disk. Use only on disks that contain a dual port kit and have been dismantled.

**/ERROR\_LOGGING**

**/NOERROR\_LOGGING**

Logs device errors in the error log file.

**/LOG**

**/NOLOG (default)**

Displays log information at the terminal.

## SET DEVICE/ACL

See Appendix ACL.

## SET DIRECTORY directory-spec,...

Modifies the characteristics of one or more directories.

### PARAMETERS

#### **directory-spec**

Disk device (optional) and name of the directory or subdirectory. Wildcard characters are allowed.

### QUALIFIERS

#### **/BACKUP**

#### **/CREATED (default)**

#### **/EXPIRED**

#### **/MODIFIED**

Selects files according to the dates of their most recent backups, their creation dates, their expiration dates, or the dates of their last modifications. Relevant only with the /BEFORE and /SINCE qualifiers.

#### **/BEFORE[=time]**

Selects only those directories with dates that precede the specified time. You can specify time as an absolute time, as a combination of absolute and delta times, or as one of the following keywords: TODAY (default), TOMORROW, or YESTERDAY. Specified with /BACKUP, /CREATED (default), /EXPIRED, or /MODIFIED.

#### **/BY\_OWNER[=uic]**

Selects only those directories with the specified user identification code. The default UIC is that of the current process.

#### **/CONFIRM**

#### **/NOCONFIRM (default)**

Issues a request for confirmation before each SET DIRECTORY operation. The following responses are valid:

YES	Modify the directory
NO	Do not modify the directory
TRUE	Modify the directory
FALSE	Do not modify the directory

## DCL-128    **DCL Commands**

### **SET DIRECTORY**

1	Modify the directory
0	Do not modify the directory
RETURN	Do not modify the directory
ALL	Continue execution of the command with no further confirmation prompts
CTRL/Z	Stop execution of the command
QUIT	Stop execution of the command

#### **/CREATED**

See /BACKUP.

#### **/EXCLUDE=(file-spec,...)**

Excludes the specified directory from modification. Wildcard characters are supported for directory specifications. You cannot include a device name.

#### **/EXPIRED**

See /BACKUP.

#### **/LOG**

#### **/NOLOG (default)**

Displays the directory specification of each directory modified as the command executes.

#### **/MODIFIED**

See /BACKUP.

#### **/OWNER\_UIC=[uic]**

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

Specifies an owner UIC for the directory. The default is the UIC of your process.

#### **/SINCE[=time]**

Selects only those directories dated after the specified time. You can specify time as an absolute time, a combination of absolute and delta times, or as one of the following keywords: TODAY (default), TOMORROW, or YESTERDAY. Specified with /BACKUP, /CREATED (default), /EXPIRED, or /MODIFIED.

#### **/VERSION\_LIMIT[=limit]**

Specifies the maximum number of versions for files with the same name and type in the directory. The value 0 implies the Files-11 architectural limit of 32,767. If you change the version limit for the directory, the new limit applies only to files created after the change has been made.

## SET DIRECTORY/ACL

See Appendix ACL.

## SET FILE file-spec,...

Modifies file characteristics.

### PARAMETERS

#### **file-spec**

A valid file specification. Wildcards are allowed.

### QUALIFIERS

#### **/BACKUP**

#### **/NOBACKUP (default)**

Specifies that BACKUP will record the contents of the file. The /NOBACKUP qualifier causes BACKUP to record the attributes of the file but not its contents. Valid only for Files-11 Structure Level 2 files.

#### **/BEFORE[=time]**

Selects only those files with dates that precede the specified time. You can specify time as an absolute time, as a combination of absolute and delta times, or as one of the following keywords: TODAY (default), TOMORROW, or YESTERDAY.

#### **/BY\_OWNER[=uic]**

Selects only those files with the specified user identification code. The default UIC is that of the current process.

#### **/CONFIRM**

#### **/NOCONFIRM (default)**

Issues a request for confirmation before each SET FILE operation. The following responses are valid:

YES	Modify the file
NO	Do not modify the file
TRUE	Modify the file
FALSE	Do not modify the file
1	Modify the file
0	Do not modify the file
RETURN	Do not modify the file

**DCL-130     DCL Commands**  
**SET FILE**

ALL	Continue execution of the command with no further confirmation prompts
CTRL/Z	Stop execution of the command
QUIT	Stop execution of the command

**/CREATED**

Selects files based on the dates of their creation. Relevant only with the /BEFORE or /SINCE qualifiers.

**/DATA\_CHECK=[([NO]READ),([NO]WRITE)]**

Verifies read operations, by rereading each record, and write operations, by reading each record after it is written. By default, a WRITE data check is performed.

**/END\_OF\_FILE**

Resets the end-of-file mark to the highest block allocated.

**/ENTER=new-file-spec**

Assigns an additional name to a single file. Use with caution. To remove one of the log names of a file, use the /REMOVE qualifier.

**/ERASE\_ON\_DELETE**

Specifies that the specified files will be physically removed from the disk when purged or deleted.

**/EXCLUDE=(file-spec,...)**

Excludes the specified file from the SET FILE operation. Wildcard characters are supported for file specifications. However, you cannot use relative version numbers to exclude a specific version. The file specification can contain a directory specification, but not a device specification.

**/EXPIRATION\_DATE=date**

**/NOEXPIRATION\_DATE**

Requires ownership of the file or access control.

Controls whether an expiration date is assigned to the specified files.

**/EXTENSION=[-n]**

Sets the extend quantity default for the file. The value of *n* can range from 0 through 65,535. A specification of /EXTENSION or /EXTENSION=0 means a system-calculated value.



**/GLOBAL\_BUFFER=n**

Sets the global buffer count for the file (the number of buffers that can be shared by processes accessing the file). The value must be an integer in the range 0 through 32,767. The value 0 disables buffer sharing.

**/LOG****/NOLOG (default)**

Displays the file specification of each file modified as the command executes.

**/NODIRECTORY**

Removes the directory attributes of a file. Valid only for Files-11 Structure Level 2 files. Use with extreme caution.

**/OWNER\_UIC=[uic]**

Requires GRPPRV for a UIC in the same group.

Requires SYSPRV for any UIC outside your group.

Specifies an owner UIC for the file. The default is the UIC of your process.

**/PROTECTION[=(ownership[:access],...)]**

Specifies protection for the specified file. The ownership categories are SYSTEM, OWNER, GROUP, and WORLD. The access categories are R (read), W (write), E (create), and D (delete). If you specify /PROTECTION without the ownership and access code, the file protection is set according to the current default protection.

**/REMOVE**

Removes one of the names of a file specified with /ENTER=new-file-spec. Use with caution.

**/SINCE[=time]**

Selects only those files dated after the specified time. You can specify time as an absolute time, as a combination of absolute and delta times, or as one of the following keywords: TODAY (default), TOMORROW, or YESTERDAY.

**/TRUNCATE**

Truncates the file at the end-of-file marker, that is, releases allocated but unused blocks of the file.

**/UNLOCK**

Makes an improperly closed file accessible.

**/VERSION\_LIMIT=[n]**

Specifies the maximum number of versions for the specified file. The value 0 implies the architectural limit of 32,767. When you exceed the limit, the earliest file is deleted without notification to the user.

## **SET FILE/ACL**

See Appendix ACL.

## **SET HOST [node-name]**

Requires DECnet if the /DTE qualifier is not specified.

Connects your terminal to another (remote) system. After the connection is made, you must observe the login procedures on the host system.

### **PARAMETERS**

#### **node-name**

The name of the remote system to which you will connect; incompatible with the /DTE qualifier.

### **QUALIFIERS**

#### **/DIAL=(NUMBER:number[,MODEM\_TYPE:modem-type])**

Allows a modem attached to the outgoing terminal line to be autodialed using the autodial protocol of that modem. *Number* is the phone number to be autodialed; *modem-type* is the type of modem being used. The default for *modem-type* is DF03. Use the /DIAL qualifier only in conjunction with the /DTE qualifier.

#### **/DTE terminal-line**

Allows you to connect your system to another (remote) system via an outgoing terminal line (rather than through a network). Specify *terminal-line* as the outgoing terminal line that connects your system either directly to the other system or to a modem. When connecting directly to another system, it is recommended that the outgoing port be set to NOTYPEAHEAD. To exit from the remote node, type CTRL/\; that is, type a backslash (\) while pressing the CTRL key.

#### **/LOG[=file-spec]**

#### **/NOLOG (default)**

Keeps a record of the entire session in the specified file. The default file specification is SETHOST.LOG. The use of this qualifier is not recommended for the purpose of file transfers.

## **SET KEY**

Sets and locks the key definition state for keys defined with the DEFINE/KEY command.

**QUALIFIERS**

**/LOG (default)**

**/NOLOG**

Displays a message indicating the key definition state that has been set.

**/STATE=state-name**

**/NOSTATE**

Specifies the name of the state.

**SET LOGINS/INTERACTIVE**

Sets the interactive limit (number of interactive users allowed on the system), or displays the interactive limit and the current number of interactive users. OPER privilege is required to log in to a system if the current number of interactive users equals or exceeds the interactive limit. Users logged in to the system are not affected by any change in the interactive limit.

**QUALIFIERS**

**/INTERACTIVE[=n]**

If *n* is specified, set the interactive limit to *n*.

If *n* is not specified, displays the interactive limit and the number of interactive users.

**SET MAGTAPE**

Defines the default characteristics associated with a specific magnetic tape device for subsequent file operations.

**PARAMETER**

**device-name**

Specifies the name of the magnetic tape device for which the characteristics are to be set. The device must not be currently allocated to any other user.

**QUALIFIERS**

**/DENSITY=density**

The **/DENSITY** qualifier is not applicable to the TK50 tape device.

Specifies the default density, in bits per inch (bpi), for all write operations on the magnetic tape device when the volume is mounted as a foreign tape or as an unlabeled tape. The density can be specified as 800, 1600, or 6250, if the density is supported by the magnetic tape drive.

## **DCL-134    DCL Commands**

### **SET MAGTAPE**

#### **/END\_OF\_FILE**

Writes a tape mark at the current position on the magnetic tape volume.

#### **/LOG**

#### **/NOLOG**

Displays information about the operations performed on the magnetic tape volume.

#### **/LOGSOFT (default)**

#### **/NOLOGSOFT**

Controls whether soft errors on the specified device are to be logged in the error log file. Soft errors are errors that are corrected by the hardware without software intervention. This qualifier only affects devices that support hardware error correction, such as the TU78 magnetic tape drive. When used with other devices, this qualifier has no effect.

#### **/REWIND**

Requests that the volume on the specified device be rewound to the beginning of the magnetic tape.

#### **/SKIP=option**

Requests that the magnetic tape volume be positioned according to any of the following options:

BLOCK:n	Directs the SET MAGTAPE command to skip the specified number of blocks ( <i>n</i> )
END_OF_TAPE	Directs the SET MAGTAPE command to position the volume at the end-of-tape mark
FILES:n	Directs the SET MAGTAPE command to skip the specified number of files ( <i>n</i> )
RECORD:n	Directs the SET MAGTAPE command to skip the specified number of records ( <i>n</i> )

#### **/UNLOAD**

Requests that the volume on the specified device be rewound and unloaded.

### **SET MESSAGE [file-spec]**

Sets the format for system messages or specifies a process level message file.

## PARAMETERS

### **file-spec**

Name of the process level message file. Messages in this file supersede messages for the same conditions in the system message file or in an existing process message file. The file type defaults to EXE. No wildcard characters are allowed. If this parameter is not specified, the qualifiers apply to the system message file.

## QUALIFIERS

### **/DELETE**

Removes any process permanent message files currently in effect. Do not specify *file-spec* with this qualifier.

### **/FACILITY**

### **/NOFACILITY**

Formats messages so that the facility name prefix appears.

### **/IDENTIFICATION**

### **/NOIDENTIFICATION**

Formats messages so that the message identification prefix appears.

### **/SEVERITY**

### **/NOSEVERITY**

Formats messages so that the severity level appears.

### **/TEXT**

### **/NOTEXT**

Formats messages so that the message text appears.

## **SET [NO]ON**

SET ON enables error checking at the current command level. Specify SET NOON to disable error checking. (Even with SET NOON, the proper values are still placed in \$STATUS and \$SEVERITY.)

## **SET OUTPUT\_RATE [=delta-time]**

Sets the rate at which output is written to a batch job log file.

PARAMETERS

**delta-time**

The time interval at which output will be written to the batch job log file. If no delta time is specified, the information is written to the log file but the output rate is not changed from the default of once per minute. Specify *delta-time* as dd-hh:mm:ss.ss.

**SET PASSWORD**

Establishes, changes, or removes a password. You will be prompted for the old password—type the password (it is not echoed) or just press RETURN if no password is established. You will then be prompted for the new password and a verification—type the new password each time or just press RETURN if you are removing the password. A password is 1 through 31 alphanumeric, dollar sign, or underscore characters (uppercase and lowercase characters are equivalent).

QUALIFIERS

**/GENERATE[=value]**

Generates a list of 5 passwords for you to select from. Press RETURN to repeat the procedure until a suitable password appears. *Value* restricts the length of the password (value is a number from 1 to 10).

**/SECONDARY**

Creates or allows you to replace a secondary password. The procedure is the same as setting your primary password. Incompatible with the /SYSTEM qualifier.

**/SYSTEM**

Requires CMKRNL and SECURITY privileges.

Changes the system password, rather than a user password.

**SET PRINTER printer-name**

Requires OPER privilege. Requires LOG\_IO privilege if the printer is spooled.

Describes the characteristics of a printer. A characteristic changes only if the qualifier is specified; otherwise, it remains the same. (The following defaults are the defaults for an initially bootstrapped system.)



PARAMETERS

**printer-name**

Device name of the printer.

QUALIFIERS

**/CR**

**/NOCR (default)**

Signals a carriage return to printers that do not include the carriage return operation as part of a line feed or vertical tab.

**/FALLBACK**

**/NOFALLBACK (default)**

Signals the printer to try to translate multinational characters into 7-bit equivalent representations. If unable, the printer will print an underscore character.

**/FF (default)**

**/NOFF**

Signals the printer to perform mechanical form feeds.

**/LA11**

Specifies the printer as an LA11.

**/LA180**

Specifies the printer as an LA180.

**/LOG**

**/NOLOG (default)**

Displays information confirming the printer setting.

**/LOWERCASE**

**/NOLOWERCASE (default)**

Passes lowercase characters to the printer. /LOWERCASE is equivalent to /NOUPPERCASE.

**/LP11**

Specifies the printer as an LP11.

**/PAGE=lines-per-page**

Specifies the number of lines per page on the print forms being used. The value must be an integer in the range 0 through 255 and defaults to 64. (The printer driver and print symbiont both use this value to determine when to perform and simulate form feeds.)

**DCL-138     DCL Commands**  
**SET PRINTER**

**/PASSALL**

**/NOPASSALL (default)**

Passes all data to the printer as 8-bit binary data (without interpreting special characters as blanks, line feeds, etc.). /PASSALL is equivalent to /NOPRINTALL.

**/PRINTALL**

**/NOPRINTALL (default)**

Expands tab characters to blanks, fills carriage return and line feed characters, and interprets control characters. /PRINTALL is equivalent to /NOPASSALL.

**/TAB**

**/NOTAB (default)**

Controls whether the printer interprets special characters or whether they are passed on to the hardware controller.

**/TRUNCATE (default)**

**/NOTRUNCATE**

Controls whether the printer truncates data exceeding the value specified by the /WIDTH qualifier. Note that the /TRUNCATE and the /WRAP qualifiers are incompatible.

**/UNKNOWN**

Specifies the printer as nonstandard.

**/UPPERCASE (default)**

**/NOUPPERCASE**

Passes only uppercase characters to the printer. /UPPERCASE is equivalent to /NOLOWERCASE.

**/WIDTH=characters-per-line**

Specifies the number of characters per line. The value must be an integer in the range 0 through 65535 and defaults to 132.

**/WRAP**

**/NOWRAP (default)**

Generates a carriage return and line feed when the value of /WIDTH is reached.

**SET PROCESS process-name**

Changes the execution characteristics associated with the specified process for the current terminal session or job.

## PARAMETERS

### **process-name**

Requires that you own the process or that you have GROUP privilege and that the process is in your group.

The name of the process for which the characteristics are to be changed. The process name can contain from 1 to 15 alphanumeric characters. The default is the current process. Compatible only with the /PRIORITY, /RESUME, and /SUSPEND qualifiers.

## QUALIFIERS

### **/DUMP**

### **/NODUMP (default)**

Causes the contents of the address space to be written to the file named SYS\$LOGIN:IMAGEDUMP.DMP when an image terminates due to an unhandled error.

### **/IDENTIFICATION=pid**

Requires GROUP or WORLD privilege for processes other than your own.

Specifies the process identification (PID) of the process for which characteristics are to be changed. Overrides the *process-name* parameter. Compatible only with the /PRIORITY, /RESUME, and /SUSPEND qualifiers.

### **/NAME=string**

Changes the name of the current process to a string of 1 through 15 characters.

### **/PRIORITY=n**

Requires ALTPRI privilege to set the priority higher than your base priority.

Changes the priority of the specified process.

### **/PRIVILEGES=(keyword,...)**

Requires SETPRV to enable a privilege you do not have.

Enables privileges for the process. The following table lists process privileges.

[NO]ACNT	Create processes for which no accounting messages are written
[NO]ALL	Have all privileges
[NO]ALLSPOOL	Allocate spooled devices
[NO]ALTPRI	Set priority values
[NO]BUGCHK	Make bug check error log entries
[NO]BYPASS	Bypass UIC protection
[NO]CMEXEC	Change mode to executive
[NO]CMKRNL	Change mode to kernel

## DCL-140    DCL Commands

### SET PROCESS

[NO]DETACH	Create detached processes
[NO]DIAGNOSE	Issue diagnostic I/O requests
[NO]EXQUOTA	Exceed quotas
[NO]GROUP	Control other processes in the same group
[NO]GRPNAM	Place names in the group logical name table
[NO]GRPPRV	Access group objects
[NO]LOG_IO	Issue logical I/O requests to a device
[NO]MOUNT	Issue a mount volume QIO request
[NO]NETMBX	Create a network device
[NO]OPER	Perform operator functions
[NO]PFNMAP	Create or delete sections mapped by page frame number
[NO]PHY_IO	Issue physical I/O requests to a device
[NO]PRMCEB	Create permanent common event flag clusters
[NO]PRMGBL	Create permanent global sections
[NO]PRMMBX	Create permanent mailboxes
[NO]PSWAPM	Alter swap mode
[NO]READALL	Bypass existing restrictions on reading a file
[NO]SECURITY	Perform security-related functions
[NO]SETPRV	Give higher privileges to other processes
[NO]SHARE	Assign a channel to a device, even if the channel is allocated to another process
[NO]SHMEM	Create data structures in shared memory
[NO]SYSGBL	Create system global sections
[NO]SYSLCK	Request locks on systemwide resources
[NO]SYSNAM	Place names in the system logical name table
[NO]SYSPRV	Access system objects
[NO]TMPMBX	Create a temporary mailbox
[NO]VOLPRO	Override volume protection
[NO]WORLD	Control all other processes in the system

#### **/RESOURCE\_WAIT**

#### **/NORESOURCE\_WAIT**

Enables resource wait mode so that the process waits for resources to become available. If you specify the /NORESOURCE\_WAIT qualifier, the process receives an error status code when system dynamic memory is not available or when the process exceeds one of the following resource quotas: direct I/O limit, buffered I/O limit, or buffered I/O byte count (buffer space) quota.

**/RESUME**

Allows a process suspended by a previous SET PROCESS command to resume operation.

**/SUSPEND**

**/NOSUSPEND**

Requires GROUP or WORLD privilege to use this qualifier.

Temporarily stops the process's activities. The qualifier /NOSUSPEND allows a suspended process to resume operation.

**/SWAPPING (default)**

**/NOSWAPPING**

Requires PSWAPM privilege for /NOSWAPPING.

Permits the process to be swapped.

**SET PROMPT [=string]**

Replaces the default DCL dollar sign prompt with the specified string.

**PARAMETERS**

**string**

The prompt string. Enclose the string in quotation marks if it contains spaces, special characters, or lowercase letters.

**QUALIFIERS**

**/CARRIAGE\_CONTROL (default)**

**/NOCARRIAGE\_CONTROL**

Inserts carriage return and line feed characters before the prompt string.

**SET PROTECTION [(ownership[:access],...)] file-spec,...**

Establishes the protection that limits other users' access to a file.

**PARAMETERS**

**ownership**

The ownership category — SYSTEM, OWNER, GROUP, or WORLD. Each category can be abbreviated to its first character.

**access**

An access category — R (read), W (write), E (execute), or D (delete) — to be assigned to a category of ownership. A null access specification means no access.

## **DCL-142    DCL Commands**

### **SET PROTECTION**

#### **file-spec**

Name of the affected file or files. Wildcard characters are allowed.

#### **QUALIFIERS**

##### **/CONFIRM**

##### **/NOCONFIRM (default)**

Prompts you for confirmation before changing protection on each file. Respond with a T or Y to change the protection.

##### **/LOG**

##### **/NOLOG (default)**

Displays a message for each file processed.

##### **/PROTECTION=(ownership[:access],...)**

Qualifies file-spec

Specifies protection for an individual file or group of files. Overrides the access specified after SET PROTECTION.

## **SET PROTECTION =(ownership[:access],...)/DEFAULT**

Establishes the protection to be applied by default to all files subsequently created.

#### **PARAMETERS**

##### **ownership**

The ownership category — SYSTEM, OWNER, GROUP, or WORLD. Each category can be abbreviated to its first character.

##### **access**

An access category — R (read), W (write), E (execute), or D (delete) — to be given to a specified type of owner. A null access specification means no access.

## **SET PROTECTION =(ownership[:access],...)/DEVICE device-name**

Requires OPER privilege.

Limits access to a non-file-structured device.

#### **PARAMETERS**

##### **ownership**

An ownership category — SYSTEM, OWNER, GROUP, or WORLD. Each category can be abbreviated to its first character.



**access**

An access category — R (read or allocate), W (write), L (logical I/O), and P (physical I/O) — to be assigned to a specified type of owner. A null access specification means no access.

**device-name**

Name of the non-file-structured device.

**QUALIFIERS**

**/OWNER\_UIC=[uic]**

Specifies an owner UIC for the device. The default owner is the UIC of your process.

**SET QUEUE queue-name[:]**

Requires OPER privilege or execute access to the specified queue.

Changes the current status or attributes of the specified queue.

**PARAMETERS**

**queue-name[:]**

The name of an execution queue or a generic queue.

**QUALIFIERS**

**/BASE\_PRIORITY=n**

Specifies the process base priority at which jobs are initiated from a batch queue. (You must stop and restart symbiont queues to change the symbiont priority for printer, terminal, or server queues.) The value of *n* can be from 0 through 15.

**/BLOCK\_LIMIT=(*[lower,]upper*)**

**/NOBLOCK\_LIMIT**

Restricts the size of print jobs that can be executed on a printer or terminal queue. The lower parameter specifies the minimum number of blocks that will be accepted by the queue for a print job. The upper parameter specifies the maximum number of blocks that will be accepted by the queue for a print job. If a job contains fewer blocks than the number specified by the lower parameter or more blocks than the number specified by the upper parameter, the job remains pending until the block limit for the queue is changed. To specify only the lower parameter, you must use two sets of quotation marks ("" ) in place of the upper specifier.

**/CHARACTERISTICS=(*characteristic,...*)**

**/NOCHARACTERISTICS**

Specifies one or more characteristics for processing jobs on the queue. Use the SHOW QUEUE/CHARACTERISTICS command to display queue characteristics.

## DCL-144    DCL Commands

### SET QUEUE

The queue must have all the characteristics specified for a job or the job remains pending.

#### **/CPUDEFAULT=time**

Specifies the default CPU time limit for batch jobs. Time can be specified as delta time, 0, NONE, or INFINITE. Both the value 0 and the keyword INFINITE allow unlimited CPU time (subject to the restrictions imposed by the /CPUMAXIMUM qualifier or the user authorization file). The keyword NONE specifies that no time limit is needed.

#### **/CPUMAXIMUM=time**

Specifies the maximum CPU time limit for batch jobs. The /CPUMAXIMUM qualifier overrides the time limit specified in the user authorization file (UAF). Time can be specified as delta time, 0, NONE, or INFINITE. Both the value 0 and the keyword INFINITE allow unlimited CPU time (subject to the restrictions imposed by the /CPUMAXIMUM qualifier or the user authorization file). The keyword NONE specifies that no time limit is needed.

#### **/DEFAULT=(option,...)**

#### **/NODEFAULT**

Sets defaults for the following PRINT options so that you do not have to specify them with the PRINT commands.

[NO]BURST[=keyword]

Specifies where to print burst pages (flag pages that are printed over the paper's perforations for easy identification of individual files in a print job.) The keyword ALL places burst pages before each printed file in the job. The keyword ONE places a burst page before the first printed file in the job.

[NO]FEED

Specifies whether a form-feed is automatically inserted at the end of a page.

[NO]FLAG[=keyword]

Specifies where to print flag pages (containing the job entry number, the name of the user submitting the job, and so on). The keyword ALL places flag pages before each printed file in the job. The keyword ONE places a flag page before the first printed file in the job.

**FORM=type**

Specifies the default form for a printer, terminal, or server queue. If a job is not submitted with an explicit form definition, then this form will be used to process the job. The systemwide default form, `form=0`, is the default value for this keyword. See also `/FORM_MOUNTED`.

**[NO]TRAILER[=keyword]**

Specifies where to print trailer pages. The keyword `ALL` places trailer pages after each printed file in the job. The keyword `ONE` places a trailer page after the last printed file in the job.

If you specify any of the keywords `BURST`, `FLAG`, `TRAILER` without specifying a value, the value `ALL` is used by default.

**/DISABLE\_SWAPPING**  
**/NODISABLE\_SWAPPING**

Allows swapping of the batch jobs.

**/ENABLE\_GENERIC**  
**/NOENABLE\_GENERIC**

Allows files queued to a generic queue that does not specify explicit queue names to be placed in this execution queue for processing.

**/FORM\_MOUNTED=type**

Specifies the form type for a printer, terminal, or server queue. (Form types are installation specific and are defined as numeric values or form names with the `DEFINE/FORM` command. The `SHOW QUEUE/FORM` command displays the available print forms.) If the stock of the mounted form is not identical to the stock of the default form, as indicated by the DCL command qualifier `/DEFAULT=FORM=type`, then all jobs submitted to this queue without an explicit form definition will enter a pending state. If a job is submitted with an explicit form and the stock of the explicit form is not identical to the stock of the mounted form, then the job will enter a pending state. In both cases, the pending state will be maintained until the stock of the mounted form of the queue is identical to the stock of the form associated with the job.

**/JOB\_LIMIT=n**

Specifies the number of batch jobs that can be executed concurrently from the queue. The value of *n* defaults to 1.

**/OWNER\_UIC=uic**

Requires `OPER` privilege.

Specifies a UIC for the queue.

**DCL-146     DCL Commands**  
**SET QUEUE**

**/PROTECTION=(ownership[:access],...)**

Requires OPER privilege.

Applies the specified protection to the queue. The ownership categories are SYSTEM, OWNER, GROUP, and WORLD. The access categories are R (read), W (write), E (create), and D (delete). The default protection is (SYSTEM:E, OWNER:D, GROUP:R, WORLD:W).

**/RECORD\_BLOCKING**

**/NORECORD\_BLOCKING**

Determines whether the symbiont can concatenate (or block together) output records for transmission to the output device. If you specify /NORECORD\_BLOCKING, the symbiont is directed to send each formatted record in a separate I/O request to the output device. For the standard MicroVMS print symbiont, record blocking can have a significant performance advantage over single-record mode.

**/RETAIN[=keyword]**

**/NORETAIN**

Retains jobs in the queue in a completed status after they have executed. Possible keywords are

ALL	Retains all jobs in the queue after execution (default)
ERROR	Retains in the queue only jobs that complete unsuccessfully

**/SCHEDULE=SIZE**

**/SCHEDULE=NOSIZE**

Specifies whether pending jobs in a printer or terminal queue are scheduled for printing based on the size of the job. When /SCHEDULE=SIZE is in effect, shorter jobs will print before longer ones.

**/SEPARATE=(keyword,...)**

**/NOSEPARATE**

Specifies the job separation defaults for a printer or terminal queue. Possible keywords are

[NO]BURST	Prints a burst page (a flag page printed over the paper's perforations for easy identification of individual files) at the beginning of every job.
[NO]FLAG	Prints a flag page (containing the job entry number, the name of the user submitting the job, and so on) at the beginning of every job.
[NO]TRAILER	Prints a trailer page at the end of every job.

**[NO]RESET=(m,...)**

Specifies a job reset sequence for the queue. The specified modules from the device control library (see **/LIBRARY**) are used to reset the device each time a job reset occurs.

**/WSDEFAULT=n**

Defines a working set default for a batch job. The **/WSDEFAULT** qualifier overrides the working set size specified in the user authorization file. Possible values are: a positive integer in the range 1 through 65,535, 0, or the keyword **NONE** (the default). The value 0 or the keyword **NONE** sets the default value to the value specified either in the UAF or by the **SUBMIT** command (if specified).

**/WSEXTENT=n**

Defines a working set extent for the batch job. The **/WSEXTENT** qualifier overrides the working set extent in the user authorization file. Possible values are: a positive integer in the range 1 through 65,535, 0, or the keyword **NONE** (the default). A zero or **NONE** sets the default value to the value specified either in the UAF or by the **SUBMIT** command (if specified).

**/WSQUOTA=n**

Defines a working set page size (working set quota) for the batch job. The **/WSQUOTA** qualifier overrides the value in the user authorization file. Possible values are: a positive integer in the range 1 through 65,535, 0, or the keyword **NONE** (the default). A zero or **NONE** sets the default value to the value specified either in the UAF or by the **SUBMIT** command (if specified).

**SET QUEUE/ENTRY =job-number queue-name**

Changes the status of a queued print or batch job that is not currently executing.

**PARAMETERS**

**job-number**

The number of the job to be executed. The job number is displayed at the time of the job's submission.

**queue-name**

Name of the print or batch queue.

**DCL-148     DCL Commands**  
**SET QUEUE/ENTRY**

**QUALIFIERS**

**/AFTER=time**

**/NOAFTER**

Holds the job until the specified time. If the time has passed, queues the job for immediate execution. The time must be in absolute date/time format.

**/BURST**

**/NOBURST**

Prints a burst page (a flag page printed over the perforations between pages for easy identification of individual files) at the beginning of the job according to one of the following keywords:

ALL            All printed files contain a burst page

ONE            The first printed file contains a burst page

**/CHARACTERISTICS=(characteristic,...)**

**/NOCHARACTERISTICS**

Specifies one or more characteristics for processing jobs on the queue. Use the SHOW QUEUE/CHARACTERISTIC command to display queue characteristics. The queue must have all the characteristics specified for a job or the job remains pending.

**/CLI=filename**

Specifies the name of a command language interpreter (CLI) to use in processing the job. The file must be in SYS\$SYSTEM and be of type EXE.

**/COPIES=n**

Specifies the number of copies to print for all files in the job. The value of *n* can be from 1 to 255.

**/CPUTIME=keyword**

Specifies a CPU time limit for the batch job. Time can be specified as: delta time, 0, NONE, or INFINITE. Both the value 0 and the keyword INFINITE allow unlimited CPU time; the keyword NONE defaults to your UAF value or the limit specified on the queue. You cannot specify more time than permitted by the base queue limits or your own UAF.

**/ENTRY=job-number**

Specifies the system-assigned number of the job. This qualifier is required and must precede all other qualifiers and parameters.

**/FEED**

**/NOFEED**

Inserts form feeds in print jobs.



**/FLAG[=keyword]**

**/NOFLAG**

Prints a flag page (containing the job entry number, the name of the user submitting the job, and other information about the file being printed) according to one of the following keywords:

ALL	Prints a flag page before each file in the job
ONE	Prints a flag page before the first file in the job

**/FORM=type**

Specifies special print form (including width, length, or type of paper) for the job. The SHOW QUEUE/FORM command displays the available print form. If you specify a form type different from that of the queue, the job remains pending until the queue's form type is set to that of the job's specified form type. To change the form type for the queue, stop the queue, physically change the form, and restart the queue, specifying the new form type with the /FORM qualifier.

**/HEADER**

**/NOHEADER**

Prints a heading line at the top of each page in the print job.

**/HOLD**

**/NOHOLD**

Holds the job until released by the /NOHOLD or /RELEASE qualifier. The /NOHOLD qualifier also releases jobs held in a queue with the /RETAIN qualifier and jobs refused by a user-written symbiont.

**/JOB\_COUNT=n**

Prints the job *n* times. The value of *n* can be from 1 through 255. This qualifier overrides both the /JOB\_COUNT qualifier specified and the default of the PRINT command.

**/KEEP**

**/NOKEEP**

Retains the batch job log file after it is printed.

**/LOG\_FILE=file-spec**

**/NOLOG\_FILE**

Creates a log file with the specified file specification, which can include a device name as long as the process executing the batch job has access to the device on which the log file will reside. Logical names in the file specification are translated in the context of the process that executes the SET QUEUE/ENTRY command. If this qualifier is not specified, a log file with the file name of the first command file (or the file name specified with the /NAME qualifier) and a file type of LOG is created.

**DCL-150    DCL Commands**  
**SET QUEUE/ENTRY**

**/LOWERCASE**  
**/NOLOWERCASE**

Prints the job only on a printer that supports lowercase characters.

**/NAME=job-name**

Names (or renames) the job. The name can be 1 through 39 alphanumeric characters. The file name defaults to the file name of the first file in the job and a file type of LOG.

**/NOCHECKPOINT**

For a batch job, erases the value established by the most recently executed SET RESTART\_VALUE command. For a print job, clears the stored checkpoint so that the job will restart from the beginning.

**/NODELETE**

Cancels file deletion for a job that was submitted with the /DELETE qualifier. If no /DELETE qualifier was specified when the job was originally submitted to the queue, you cannot use the SET QUEUE/ENTRY to establish file deletion at a later time. You cannot use the /NODELETE qualifier to specify that individual files in a multi file job not be deleted.

**/NOTE=message**

Specifies a message of up to 255 characters to appear on the flag page of the job. Enclose the message in quotation marks if it contains spaces, special characters, or lowercase characters.

**/NOTIFY**  
**/NONOTIFY**

Broadcasts notification of job completion or abortion to any terminal at which you are logged in.

**/OPERATOR=message**

Specifies a message string of up to 255 characters to be sent to the operator just before the job begins execution. Enclose the message in quotation marks if it contains spaces, special characters, or lowercase characters.

**/PAGES=([lower],[upper])**

Specifies the number of pages to print for the specified job. By default, all pages of a file are printed. The lower parameter specifies the first page to print; it defaults to the first page of the job. The upper parameter specifies the last page to print; it defaults to the last page of the job. You must specify double quotation marks (" ") if you omit the upper parameter.

**/PARAMETERS=(parameter,...)**

Specifies from 1 to 8 optional parameters to be passed to the job. A parameter can consist of 1 through 255 characters. Enclose the parameter in quotation marks if it contains spaces, special characters, or lowercase characters. For batch jobs, the parameters define values to be equated to the symbols named P1 through P8 in each command procedure in the job.

**/PASSALL**

**/NOPASSALL**

Specifies whether the symbiont bypasses all formatting and sends the output QIO to the driver with format suppressed. All qualifiers affecting formatting, as well as the /HEADER, /PAGES, and /PAGE\_SETUP qualifiers, will be ignored.

When you use the /PASSALL qualifier with the SET QUEUE/ENTRY command, the qualifier applies to the entire job. You cannot use this qualifier to specify PASSALL mode for individual files within a multifile job.

**/PRINTER[=queue-name]**

**/NOPRINTER**

Queues the batch job log to the specified printer queue when the job is completed. By default, the printer queue for the log file is SYS\$PRINT. The /NOPRINTER qualifier assumes the /KEEP qualifier.

**/PRIORITY=n**

Requires OPER or ALTPRI privilege to raise the priority above the value of the SYSGEN parameter MAXQUEPRI.

Specifies the base priority at which a batch job will execute. The value of *n* can be from 0 to 31 and defaults to the value of the SYSGEN parameter DEFQUEPRI.

**/RELEASE**

Releases for processing jobs submitted with the /AFTER qualifier, jobs held in a queue with the /RETAIN qualifier, and jobs refused by a user-written symbiont.

**/REQUEUE=queue-name**

Moves the job to the specified queue.

**/RESTART**

**/NORESTART**

Restarts a batch or print job after a system crash or a STOP/REQUEUE command.

**/SETUP=module,...**

Extracts the specified module from the device control library (containing escape sequence modules for programmable printers) and copies the module to the printer before a file is printed.

## **DCL-152    DCL Commands**

### **SET QUEUE/ENTRY**

**/SPACE**  
**/NOSPACE**

Double spaces a print job. The default is single spacing.

**/TRAILER[=keyword]**  
**/NOTRAILER**

Prints a trailer page (containing the job entry number, the name of the user submitting the job, and other information about the file) according to one of the following keywords:

ALL	All printed files contain a trailer page
ONE	The last printed file contains a trailer page

**/WSDEFAULT=n**

Defines a working set default for a batch job. The /WSDEFAULT qualifier overrides the working set size specified in the user authorization file. Possible values are: a positive integer in the range 1 through 65,535, 0, or the keyword NONE (the default). A value of 0 or the keyword NONE sets the default value to the value specified either in the UAF or by the SUBMIT command (if specified).

**/WSEXTENT=size**

Defines a working set extent for the batch job. The /WSEXTENT qualifier overrides the working set extent in the user authorization file. Possible values are: a positive integer in the range 1 through 65,535, 0, or the keyword NONE (the default). A zero or NONE sets the default value to the value specified either in the UAF or by the SUBMIT command (if specified).

**/WSQUOTA=size**

Defines a working set page size (working set quota) for the batch job. The /WSQUOTA qualifier overrides the value in the user authorization file. Possible values are: a positive integer in the range 1 through 65,535, 0, or the keyword NONE (the default). A zero or NONE sets the default value to the value specified either in the UAF or by the SUBMIT command (if specified).

## **SET RESTART\_VALUE=string**

Sets a value for the symbol BATCH\$RESTART.

PARAMETERS

**string**

A string of up to 255 characters specifying the label at which the batch job should begin executing again.

## SET RIGHTS\_LIST id-name

Modifies the process or system rights list. You must specify either /DISABLE or /ENABLE with the SET RIGHTS\_LIST command.

### PARAMETER

#### id-name[,...]

Identifiers to be added to or removed from the process or system rights list.

**Id-name** is a string of 1 to 31 alphanumeric characters, underscores, and dollar signs. At least one character must be nonnumeric.

### QUALIFIERS

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

Specifies attributes to be added to new or existing identifiers. Valid keywords are:

[NO]DYNAMIC Indicates whether or not unprivileged holders of the identifiers may add or remove them from the process rights list. The default is NODYNAMIC.

[NO]RESOURCE Indicates whether or not holders of the identifiers may charge resources to them. The default is NORESOURCE.

#### /ENABLE

#### /DISABLE

Adds or removes the identifiers to or from the process or system rights list.

#### /IDENTIFICATION=pid

#### /PROCESS[=process-name]

Requires CMKRNL privilege and GROUP or WORLD privilege to affect other processes on the system.

Identifies the process identification value (pid) or name of the process whose rights list is to be modified. The process name can contain from 1 to 15 alphanumeric characters. The default is the current process. You cannot use either of these qualifiers with the /SYSTEM qualifier.

#### /SYSTEM

Requires CMKRNL and SYSNAM privilege.

Specifies that the desired operation be performed on the system rights list. Incompatible with /PROCESS or /IDENTIFICATION.



## **SET RMS\_DEFAULT**

Defines default values for the multiblock and multibuffer counts used by MicroVMS RMS for file operations. Defaults are set for sequential, indexed-sequential, or relative access files on a process-only basis, unless a systemwide basis is requested. For indexed sequential files, SET RMS\_DEFAULT defines default prologue level options. For sequential files, SET RMS\_DEFAULT defines default extensions; if your program does not specify a default extension, the process or system default is used.

### **QUALIFIERS**

#### **/BLOCK\_COUNT=count**

Specifies a default multiblock count (from 0 through 127) for file operations, where *count* is the number of blocks to be allocated for each I/O buffer.

#### **/BUFFER\_COUNT=count**

Specifies a default multibuffer count (from 0 through 127) for file operations, where *count* is the number of buffers to be allocated. If you do not specify the type of files (/DISK, /INDEXED, /RELATIVE, /SEQUENTIAL, and /UNIT\_RECORD) to which the default is to be applied, /SEQUENTIAL is assumed.

#### **/DISK**

Applies the specified defaults to file operations on disks. (The /SEQUENTIAL qualifier assumes /DISK.)

#### **/EXTEND\_QUANTITY=n**

Specifies the number of blocks (*n*) to extend a sequential file; the value of *n* can be from 0 to 65535. If you do not specify /EXTEND\_QUANTITY, MicroVMS RMS calculates its own extend value.

#### **/INDEXED**

Applies the specified defaults to indexed file operations.

#### **/MAGTAPE**

Indicates that the specified multibuffer default is to be applied to operations on magnetic tape volumes. If /SEQUENTIAL is specified, /MAGTAPE is assumed.

#### **/NETWORK\_BLOCK\_COUNT=count**

Specifies a default block count for network access to remote sequential, indexed sequential, and relative files. Specify *count* as a value in the range of 0 to 127.

#### **/PROLOG=n**

Specifies a default prologue level for indexed sequential files where *n* is 0, 2, or 3. (A value of 1 is not allowed.) By default, the value of *n* is 0; if *n* is 0, the record management system (RMS) sets an appropriate prologue level.



**/RELATIVE**

Applies the specified defaults to relative file operations.

**/SEQUENTIAL (default)**

Applies the specified defaults to sequential file operations.

**/SYSTEM**

Requires the change-mode-to-kernel (CMKRNL) privilege.

Indicates that the specified defaults are to be applied to file operations by all processes.

**/UNIT\_RECORD**

Applies the specified defaults to file operations on unit record devices.

**SET SYMBOL**

Controls access to local and global symbols in command procedures.

**QUALIFIER**

**/SCOPE=(keyword,...)**

Controls access to local and global symbols. Allows the user to treat symbols as being undefined. Possible keywords are:

- |            |   |
|------------|---|
| [NO]LOCAL  | Specifying the NOLOCAL keyword causes all local symbols defined in outer procedure levels to be treated as being undefined by the current procedure and all inner procedure levels. Specifying LOCAL removes any symbol translation limit set by the current procedure level. |
| [NO]GLOBAL | Specifying the NOGLOBAL keyword causes all global symbols to be inaccessible to the current procedure level and all inner procedure levels unless otherwise changed. Specifying GLOBAL restores access to all global symbols.   |

Note that when you exit a procedure back to a previous procedure, the symbol scoping context from the previous level is restored for both local and global symbols.

**SET TERMINAL [device-name]**

Sets the characteristics of a terminal. Entering a qualifier changes a characteristic. Omitting a qualifier leaves the characteristic unchanged. (The /DEVICE\_TYPE qualifier sets the default characteristics for the specified terminal type; the /INQUIRE qualifier automatically assigns default characteristics according to terminal type.)

## **DCL-156    DCL Commands**

### **SET TERMINAL**

#### **PARAMETERS**

##### **device-name**

Device name of the terminal. The default is SYS\$COMMAND if that device is a terminal.

#### **QUALIFIERS**

##### **/ADVANCED\_VIDEO**

##### **/NOADVANCED\_VIDEO**

Specifies that the terminal has advanced video attributes and is capable of 132-column video. If the terminal width is set to 132 columns and /ADVANCED\_VIDEO is enabled, the terminal page limit is set to 24 lines; if /NOADVANCED\_VIDEO is enabled, the terminal page limit is set to 12 lines.

##### **/ALTYPEAHD**

##### **/NOALTYPEAHD**

Sets the size of the type-ahead buffer when used with the /PERMANENT qualifier.

##### **/ANSI\_CRT (default)**

##### **/NOANSI\_CRT**

Conforms to ANSI standards for terminal transmissions.

##### **/APPLICATION\_KEYPAD**

##### **/NUMERIC\_KEYPAD (default)**

Specifies whether the keys of the numeric keypad will be used to type numbers and punctuation marks (/NUMERIC\_KEYPAD) or to enter DCL commands defined with the DEFINE/KEY command (/APPLICATION\_KEYPAD).

##### **/AUTOBAUD**

##### **/NOAUTOBAUD**

Specifies whether the terminal baud rate is reset when you log in. You must press the RETURN key two or more times at intervals of at least one second for the baud rate to be correctly determined. If you press a key other than RETURN, /AUTOBAUD might detect the wrong baud rate. If this happens, wait for the login procedure to time out before continuing. The /AUTOBAUD qualifier must be used with the /PERMANENT qualifier.

##### **/BLOCK\_MODE**

##### **/NOBLOCK\_MODE**

Performs block mode transmission, local editing, and field protection.

##### **/BRDCSTMBX**

##### **/NOBRDCSTMBX (default)**

Sends broadcast messages to an associated mailbox if it exists.

**/BROADCAST (default)**

**/NOBROADCAST**

Enables reception of broadcast messages (such as those issued by MAIL and REPLY).

**/CRFILL[=fill-count]**

Generates the specified number of null characters after each carriage return before transmitting the next meaningful character (to ensure that the terminal is ready for reception). The value must be an integer in the range 0 through 9 and defaults to 0.

**/DEC\_CRT[=(value1,value2)]**

**/NODEC\_CRT[=(value1,value2)]**

Specifies that the terminal conforms to DEC VT100-family standards and supports the minimum VT100 standards, which include the VT100 escape sequences.

Two optional values may be specified. A value of 1 requests that the DEC\_CRT terminal characteristic be set. This is the default. A value of 2 requests that the DEC\_CRT2 terminal characteristic be set. This determines whether the terminal conforms to DEC VT200-family standards and supports the minimum VT200 standards, including additional DEC escape sequences.

Note that DEC\_CRT2 is a superset of DEC\_CRT. Clearing DEC\_CRT will cause DEC\_CRT2 to be cleared. Similarly, setting DEC\_CRT2 will cause DEC\_CRT (and ANSI\_CRT) to be set.

**/DEVICE\_TYPE=device-type**

Informs the system of the terminal type and sets characteristics according to the device type specified. The default characteristics for the VT100, VT102, VT125, and VT200 series terminals are as follows:

/ADVANCEDVIDEO	/CRFILL=0	/LFFILL=0	/SPEED=9600
/NOALTYPEAHD	/ECHO	/LOWERCASE	/TAB
/ANSI_CRT	/NOEIGHT_BIT	/NODMA	/TTSYNC
/NOAUTOBAUD	/NOESCAPE	/PAGE=24	/TYPE_AHEAD
/NOBLOCK_MODE	/NOFORM	/NOPARITY	/WIDTH=80
/NOBRDCSTMBX	/FULLDUP	/NOPASTHRU	/WRAP
/BROADCAST	/NOHOSTSYNC	/NOREADSYN	

**/DIALUP**

**/NODIALUP (default)**

Specifies that the terminal is a dialup terminal.

**DCL-158     DCL Commands**  
**SET TERMINAL**

**/DISCONNECT**

**/NODISCONNECT (default)**

Specifies that the process connected to this terminal not be disconnected if the line detects a hangup. The /DISCONNECT qualifier is only valid when /PERMANENT is specified.

**/DISMISS**

**/NODISMISS (default)**

Makes the terminal driver ignore data causing a parity error (instead of terminating the currently outstanding I/O with an error status).

**/DMA**

**/NODMA**

Controls the use of direct memory access (DMA) mode on a controller that supports this feature.

**/ECHO (default)**

**/NOECHO**

Makes the terminal display the input it receives. With /NOECHO, the terminal displays only system and/or user application output.

**/EDIT\_MODE**

**/NOEDIT\_MODE**

Specifies that the terminal can perform ANSI-defined advanced editing functions.

**/EIGHT\_BIT**

**/NOEIGHT\_BIT (default)**

Uses 8-bit ASCII protocol rather than 7-bit ASCII protocol.

**/ESCAPE**

**/NOESCAPE (default)**

Validates escape sequences.

**/FALLBACK**

**/NOFALLBACK**

Displays the 8-bit DEC Multinational Character Set characters on the terminal in their 7-bit representation. The default depends on the /EIGHTBIT setting of the terminal.

**/FORM**

**/NOFORM**

Transmits a form feed rather than translating it into multiple line feeds.

**/FRAME=*n***

Specifies the number of data bits that the terminal driver expects for every character that is input or output. The value of *n* can be from 5 through 8. The default value depends on the /PARITY and /EIGHTBIT settings of the terminal.

**/FULLDUP**

**/NOFULLDUP**

Operates in full duplex mode. /FULLDUP is equivalent to /NOHALFDUP.

**/HALFDUP (default)**

**/NOHALFDUP**

Operates in half duplex mode. /HALFDUP is equivalent to /NOFULLDUP.

**/HANGUP**

**/NOHANGUP (default)**

May require LOG\_IO or PHY\_IO privilege depending on system parameter settings.

Hangs up the terminal modem when you log out.

**/HARDCOPY**

**/NOHARDCOPY**

Establishes the device as a hardcopy terminal and thus outputs a backslash (\) when the DELETE key is pressed. The /HARDCOPY qualifier is equivalent to /NOSCOPE.

**/HOSTSYNC**

**/NOHOSTSYNC (default)**

When you specify the /HOSTSYNC qualifier, the system stops transmission to the terminal (by generating a CTRL/S) when the input buffer is full and resumes transmission (by generating a CTRL/Q) when the input buffer is empty.

**/INQUIRE**

Sets the device type according to a response elicited from the terminal; the default is UNKNOWN. Works only on DIGITAL terminals, and not on the LA36 or VT05 terminals.

**/INSERT**

**/OVERSTRIKE (default)**

Specifies whether you can insert a character (/INSERT) when editing command lines, or type over a character (/OVERSTRIKE). You can use CTRL/A to switch from one mode to the other.

**DCL-160     DCL Commands**  
**SET TERMINAL**

**/LFFILL[=fill-count]**

Transmits to the terminal the specified number of null characters after each line feed before transmitting the next meaningful character (to ensure that the terminal is ready for reception). The value must be an integer in the range 0 through 9.

**/LINE\_EDITING**

**/NOLINE\_EDITING (default)**

Enables advanced line-editing features for editing command lines: both RETURN and CTRL/Z are recognized as line terminators, as are escape sequences.

**/LOCAL\_ECHO**

**/NOLOCAL\_ECHO (default)**

Echoes characters locally (rather than the host echoing them) for command level terminal functions. (Do not use /LOCAL\_ECHO with utilities that require control over echoing, such as line editing or EDT's screen mode.) Note that MicroVMS cannot control the echoing of passwords when /LOCAL\_ECHO is set.

**/LOWERCASE**

**/NOLOWERCASE**

Passes lowercase characters to the terminal. The /NOLOWERCASE qualifier translates all input to uppercase. /LOWERCASE is equivalent to /NOUPPERCASE.

**/MANUAL**

Indicates manual switching of terminal lines to dynamic asynchronous DDCMP lines when your local terminal emulator does not support automatic switching. The /MANUAL qualifier should be specified with the /PROTOCOL=DDCMP and /SWITCH=DECNET qualifiers.

**/MODEM**

**/NOMODEM**

Indicates that the terminal is connected to a modem or a cable that supplies standard EIA modem control signals. If your terminal has the MODEM characteristic, typing SET TERMINAL/NOMODEM automatically logs you out.

**/NUMERIC\_KEYPAD**

See /APPLICATION\_KEYPAD.

**/OVERSTRIKE**

See /INSERT.



**/PAGE[=lines-per-page]**

For hardcopy terminals, specifies the number of print lines between perforations. (When the terminal reads a form feed, it advances the paper to the next perforation.) The value of *n* can be from 0 through 255 and defaults to 0 (which treats a form feed as a line feed).

**/PARITY[=option]**

**/NOPARITY (default)**

Passes data with odd or even parity, where *option* equals ODD or EVEN. If you specify /PARITY without an option, the value defaults to EVEN.

**/PASTHRU**

**/NOPASTHRU (default)**

Passes all data (including tabs, carriage returns, line feeds, and control characters) to an application program as binary data.

**/PERMANENT**

Requires LOG\_IO or PHY\_IO privilege.

Sets characteristics on a permanent basis, that is, over terminal sessions. However, the characteristics revert to their initial values if the system is halted and restarted. Use in a system start-up file to establish characteristics for all terminals on the system.

**/PRINTER\_PORT**

**/NOPRINTER\_PORT (default)**

Specifies that the terminal has a printer port (an attribute not set by the SET TERMINAL/INQUIRE command).

**/PROTOCOL=DDCMP**

**/PROTOCOL=NONE (default)**

Controls whether the terminal port specified is changed into an asynchronous DDCMP line. The /PROTOCOL=NONE qualifier changes an asynchronous DDCMP line back into a terminal line. Note that /PROTOCOL=DDCMP is a permanent characteristic; therefore, the /PERMANENT qualifier is not required.

**/READSYNC**

**/NOREADSYNC (default)**

Uses the CTRL/S and CTRL/Q functions to synchronize data transmitted from the terminal.

**/REGIS**

**/NOREGIS**

Specifies that the terminal understands ReGIS graphic commands.

**DCL-162    DCL Commands**  
**SET TERMINAL**

**/SCOPE**  
**/NOSCOPE**

Establishes the device as a video terminal. /SCOPE is equivalent to /NOHARDCOPY.

**/SECURE\_SERVER**  
**/NOSECURE\_SERVER (default)**

Sets the BREAK key on the terminal to log out the current process. The /SECURE\_SERVER qualifier has no effect on terminals set with /AUTOBAUD.

**/SET\_SPEED**  
**/NOSET\_SPEED**

Requires either LOG\_IO or PHY\_IO privilege.

Allows the /SPEED qualifier to be used to change the terminal speed.

**/SIXEL\_GRAPHICS**  
**/NOSIXEL\_GRAPHICS (default)**

Specifies that the terminal is capable of displaying graphics using the sixel graphics protocol.

**/SOFT\_CHARACTERS**  
**/NOSOFT\_CHARACTERS (default)**

Specifies that the terminal is capable of loading a user-defined character set.

**/SPEED=(input-rate,output-rate)**

Sets the baud rate at which the terminal receives and transmits data. If the input and output rates are the same, specify /SPEED=rate.

**/SWITCH=DECNET**

Causes the terminal lines at each node to be switched to dynamic asynchronous DDCMP lines, when specified with the /PROTOCOL=DDCMP qualifier. Note that /SWITCH=DECNET is a permanent characteristic; therefore, the /PERMANENT qualifier is not required.

**/SYSPASSWORD**  
**/NOSYSPASSWORD (default)**

Requires LOG\_IO privilege.

Determines whether the terminal requires that a system password be entered before the USERNAME prompt.

**/TAB**  
**/NOTAB (default)**

Does not convert tab characters to multiple blanks. The /NOTAB qualifier expands all tab characters to blanks and assumes tab stops at 8-character intervals.

**/TTSYNC (default)**  
**/NOTTSYNC**

Stops transmitting to the terminal when CTRL/S is pressed and resumes transmission when CTRL/Q is pressed.

**/TYPE\_AHEAD (default)**  
**/NOTYPE\_AHEAD**

Accepts unsolicited input for the terminal to the limit of the type-ahead buffer.

**/UNKNOWN**

Specifies a terminal type that is unknown to the system, which then uses the default terminal characteristics for unknown terminals.

**/UPPERCASE**  
**/NOUPPERCASE**

Passes only uppercase characters to the terminal. /UPPERCASE is equivalent to /NOLOWERCASE.

**/WIDTH=characters-per-line**

Specifies the maximum characters per line. This value must be an integer in the range 1 through 511. With /WRAP, the terminal generates a carriage return and line feed when the width specification is reached.

**/WRAP (default)**  
**/NOWRAP**

Generates a carriage return and line feed when the value of /WIDTH is reached.

**SET TIME [=time]**

Requires both OPER and LOG\_IO privileges.

Resets the system clock, which is used both as a timer to record intervals between various internal events and as a source clock for displaying the time of day.

**PARAMETERS**

**time**

A date in the format day-month-year, a time in the format hour:minute:second.hundredth, or both. *Day* must be an integer in the range 1 through 31. *Month* must be JAN, FEB, MAR, APR, MAY, JUN, JUL, AUG, SEP, OCT, NOV, or DEC. *Year* must be an integer in the range 1858 through 9999. *Hour* must be an integer in the range 0 through 23. *Minute* must be an integer in the range 0 through 59. *Second* must be an integer in the range 0 through 59. *Hundredth* (of a second) must be an integer in the range 0 through 99. The hyphens, colons, and period are required delimiters. Delimit the date and time, when both are specified, with a colon.

**DCL-164    DCL Commands**  
**SET UIC**

**SET UIC uic**

Requires CMKRNL (change mode to kernel mode) privilege.

Changes the user identification code (UIC) of your process.

**PARAMETERS**

**[uic]**

A valid UIC. Brackets are required around the UIC.

**SET [NO]VERIFY [=[([NO]PROCEDURE,[NO]IMAGE)]**

Controls whether command lines and/or data lines in command procedures are displayed at the terminal or printed in a batch job log. Specify the keyword PROCEDURE to write each DCL command line in a command procedure to the output device. Specify IMAGE to write data lines to the output device. By default, both types of verification are set or cleared with SET VERIFY and SET NOVERIFY. The default setting for command procedures executed interactively is SET NOVERIFY; the default for batch jobs is SET VERIFY.

**PARAMETERS**

**PROCEDURE (default)**

**/NOPROCEDURE**

Writes each DCL command line in a command procedure to the output device.

**IMAGE (default)**

**/NOIMAGE**

Writes data lines (input data that is included as part of the SYS\$INPUT input stream) to the output device.

**SET VOLUME device-spec[:],...**

Requires WRITE access to the index file on the volume.

Changes the characteristics of a mounted Files-11 volume.

**PARAMETERS**

**device-spec[:]**

The name of a mounted Files-11 volume.

## QUALIFIERS

### **/ACCESSED=[*n*]**

Requires OPER privilege.

Specifies the number of directories to be maintained in system space for ready access. The value of *n* can be from 0 through 255 and defaults to 3. If you specify a value greater than the current value, the new value is effective immediately; otherwise, the new value will not take effect until the next time the volume is mounted.

### **/DATA\_CHECK[=(keyword,...)]**

Defines a default for data check operations following all reads and/or writes to the specified volume. (If you do not specify the /DATA\_CHECK qualifier, no checks are made.) Possible keywords are

READ            Performs checks following all read operations

WRITE           Performs checks following all write operations (default)

### **/ERASE\_ON\_DELETE**

### **/NOERASE\_ON\_DELETE (default)**

Files on the volume are overwritten with zeros when they are deleted.

### **/EXTENSION=*n***

Sets the extend quantity default for all files on the volume. The value *n* can range from 0 through 65535.

### **/FILE\_PROTECTION [=(ownership[:access],...)]**

Sets the default protection to be applied to all files on the specified disk volume. Specify ownership as SYSTEM, OWNER, GROUP, or WORLD and access as R (read), W (write), E (execute), or D (delete). A null access specification means no access.

### **/HIGHWATER\_MARKING**

### **/NOHIGHWATER\_MARKING**

Sets the File Highwater Mark (FHM) volume attribute, which guarantees that a user cannot read data that he has not written. Applies to Structure Level 2 volumes only.

### **/LABEL=volume-label**

Specifies a 1 through 12-character alphanumeric name to be encoded on the volume. Characters are automatically changed to uppercase.

### **/LOG**

### **/NOLOG (default)**

Displays the volume specification of each volume after the modification.

**DCL-166     DCL Commands**  
**SET VOLUME**

**/MOUNT\_VERIFICATION**  
**/NOMOUNT\_VERIFICATION**

Enables mount verification (preventing interruption to user input/output operations and notifying the operator of problems with the disk).

**/OWNER\_UIC[=uic]**

Sets the owner UIC of the volume to the specified UIC. The default UIC is that of the current process. Brackets are required around the UIC.

**/PROTECTION=(ownership[:access],...)**

Specifies the protection to be applied to the volume. The ownership categories are SYSTEM, OWNER, GROUP, and WORLD; the access categories are R (read), W (write), E (create), and D (delete). The default protection is all types of access by all categories of user.

**/REBUILD**

Recovers caching limits for a volume that was improperly dismounted. If a disk volume was dismounted improperly (such as during a system failure), and was then remounted with the MOUNT/NOREBUILD command, you can use SET VOLUME/REBUILD to recover the caching that was in effect at the time of the dismount.

**/RETENTION=(minimum[,maximum])**

Specifies the minimum and maximum retention times to be used by the file system to determine the expiration date for files on the volume. When a file is created, its expiration date is set to the current time + maximum. Each time the file is accessed, the current time is added to the minimum time and if the sum is greater than the expiration date, a new expiration date is computed. Maximum defaults to the smaller of (2 x minimum) or (minimum + 7).

**/UNLOAD (default)**

**/NOUNLOAD**

Specifies for the DISMOUNT command that the volume is unloaded.

**/USER\_NAME[=username]**

Specifies a user name of up to 12 alphanumeric characters to be recorded on the volume. The default name is the current process user name.

**/WINDOWS[=n]**

Specifies the number of mapping pointers to be allocated for file windows. The value of *n* can be from 7 through 80; the default value is 7.



## **SET WORKING\_SET**

Redefines the default working set size of the current process or sets an upper limit to which the working set size can be changed by an image that the process executes. (The working set limits cannot be set to exceed those defined in the user authorization file.)

### **QUALIFIERS**

**/ADJUST (default)**

**/NOADJUST**

Enables the system's changing of the process working set.

**/EXTENT=n**

Specifies the maximum number of pages that can be resident in the working set during image execution. The extent value must be greater than the minimum working set defined at system generation and must be less than or equal to the authorized extent defined in the user authorization file. If you specify a value greater than the authorized extent, the command sets the working set limit at the maximum authorized value.

**/LIMIT=n**

Specifies the size to which the working set is to be reduced at image exit. If you specify a value greater than the current quota, the quota value is also increased. If you specify a limit equal to the **/QUOTA** value, automatic working set adjustment is disabled.

**/LOG**

**/NOLOG (default)**

Displays confirmation of the **SET WORKING\_SET** command.

**/QUOTA=n**

Specifies the maximum number of pages that any image executing in the process context can request. If you specify a quota value that is greater than the authorized quota, the working set quota is set to the authorized quota value.

## **SHOW ACCOUNTING**

Displays the activities for which accounting is currently enabled.

### **QUALIFIERS**

**/OUTPUT[=file-spec]**

**/NOOUTPUT**

Specifies the file to which the display is written; by default, the display is written to the current **SYS\$OUTPUT** device.

## **SHOW ACL**

See Appendix ACL.

## **SHOW AUDIT**

Requires the SECURITY privilege.

Displays the set of auditing features that have been enabled with the SET AUDIT command and the events that they will report.

QUALIFIERS

**/OUTPUT[=file-spec]**

**/NOOUTPUT**

Specifies the file to which the display is written; by default, the display is written to the current SYS\$OUTPUT device.

## **SHOW BROADCAST**

Displays the message classes that are currently enabled by the SET BROADCAST command.

QUALIFIERS

**/OUTPUT[=file-spec]**

**/NOOUTPUT**

Specifies the file to which the display is written; by default, the display is written to the current SYS\$OUTPUT device.

## **SHOW DEFAULT**

Displays the current default device and directory.

## **SHOW DEVICES [device-name]**

Displays information about the devices on the system.

PARAMETERS

**device-name**

Name of the device. The name can be generic — if no controller or unit number is specified, all devices that satisfy that portion of the name are displayed. For example, D means all disk devices.

**QUALIFIERS**

**/ALLOCATED**

Displays information on allocated devices.

**/BRIEF (default)**

**/FULL**

Provides a brief display or a full display. A brief display contains the following information: time, device name, status (on line or not), characteristics (if the device is allocated, if it is spooled, if it has a volume mounted on it, if it has a foreign volume mounted on it), error count, volume label, free blocks on the volume, number of transactions, number of mount requests.

A full display contains the information of a brief display plus: number of I/O operations completed, number of references, process identification and name of the device owner, default buffer size, UIC of volume owner, volume protection, volume status (if it is mounted /SYSTEM or /GROUP), name of the volume's ACP, relative volume number, default cluster size, maximum number of files allowed on the volume.

**/FILES**

Requires SYSPRV or BYPASS privilege to read protected files.

Names the open files on the volume. A blank file name indicates a temporary file. If the /SYSTEM qualifier is also specified, only the names of installed files and files opened by the system are displayed (including files opened without the ACP and system files). If /NOSYSTEM is specified, only files opened by processes are displayed. Incompatible with /ALLOCATED, /BRIEF, /FULL or /MOUNTED.

**/FULL**

See /BRIEF.

**/MOUNTED**

Displays devices with volumes mounted on them.

**/OUTPUT[=file-spec]**

**/NOOUTPUT**

Specifies the file to which the display is written; by default, the display is written to the current SYS\$OUTPUT device.

**/SYSTEM**

**/NOSYSTEM**

Names (when /FILES is specified) only those files opened by the system or only those files opened by a process (/NOSYSTEM). The default is all open files.

## **DCL-170    DCL Commands**

### **SHOW DEVICES**

#### **/WINDOWS**

Displays the window count and total size of all windows for files open on a volume, as well as the file name and related process name and process identification (PID). The letter C in the display indicates that the file is open with cathedral (segmented) windows.

#### **SHOW ERROR**

Displays any nonzero error count for the CPU, memory, and devices.

#### **QUALIFIERS**

##### **/FULL**

Displays the error count for all devices, including those with no errors.

##### **/OUTPUT[=file-spec]**

##### **/OUTPUT=SYS\$OUTPUT (default)**

Specifies the file to which the display is written; by default, the display is written to the current SYS\$OUTPUT device.

#### **SHOW INTRUSION**

Requires CMKRNL and SECURITY privileges.

Displays the contents of the breakin database.

#### **QUALIFIERS**

##### **/OUTPUT[=file-spec]**

Specifies the file to which output is written; by default, the display is written to the current SYS\$OUTPUT device.

##### **/TYPE=keyword**

Selects the type of information displayed with one of the following keywords:

ALL (default)	Displays all breakin entries.
SUSPECT	Displays breakin entries for login failures that have occurred but have not been identified as intruder.
INTRUDER	Displays breakin entries for which the login failure rate was high enough to warrant evasive action.

## SHOW KEY [key-name]

Displays key definitions created with the DEFINE/KEY command.

### PARAMETERS

#### key-name

The name of the key. Permissible keys are as follows:

Keyname	VT100 Key	VT200 Key
PF1	PF1	PF1
PF2	PF2	PF2
PF3	PF3	PF3
PF4	PF4	PF4
KP0, KP1—KP9	Keypad 0—9	Keypad 0—9
PERIOD	Period Key	Period Key
COMMA	Comma Key	Comma Key
MINUS	Minus Key	Minus Key
ENTER	Enter Key	Enter Key
FIND, INSERT HERE	-	Find, Insert Here
REMOVE, SELECT	-	Remove, Select
PREV_SCREEN	-	Prev Screen
NEXT_SCREEN	-	Next Screen
HELP, DO	-	Help(F15), Do(F16)
F6—F20	-	Function Keys F6—F20

**NOTE:** You cannot define the UP and DOWN arrow keys or function keys F1 through F5. You must issue the SET TERMINAL/NOLINE\_EDITING command before defining the LEFT and RIGHT arrow keys and function keys F6 through F14.

### QUALIFIERS

#### /ALL

Displays all key definitions in the current state (or the state specified with the /STATE qualifier), including the state for each definition and all qualifiers that are associated with each definition. Incompatible with the *key-name* parameter.

#### /BRIEF (default)

#### /FULL

Determines whether only the key definition and state are displayed (/BRIEF) or whether all qualifiers associated with the key definition are displayed as well (/FULL).

## **DCL-172    DCL Commands**

### **SHOW KEY**

#### **/DIRECTORY**

Displays the names of all states for which keys have been defined.

#### **/STATE=(state-name,...)**

#### **/NOSTATE**

Displays the key definition for the specified state. The current state is the default.

### **SHOW LOGICAL [logical-name,...]**

Displays a logical name or names, equivalences, level of translation, and logical name table. If no logical name is specified, displays the logical names in the list of tables specified by LNM\$DCL\_LOGICAL.

#### **PARAMETERS**

##### **logical-name**

A logical name or names. The asterisk (\*) and per cent (%) wildcard characters are allowed; however, if a wildcard character is present, iterative translation is not done.

#### **QUALIFIERS**

##### **/ACCESS\_MODE=mode**

Displays names defined in the specified access mode and any inner access modes. You can specify one of the following keywords to indicate the access mode: USER\_MODE, SUPERVISOR\_MODE, EXECUTIVE\_MODE or KERNEL\_MODE. The default is USER\_MODE. By default, the system displays any definitions in all three access modes.

##### **/ALL (default)**

Indicates that all logical names in the specified logical name tables are to be displayed.

##### **/DESCENDANTS**

##### **/NODESCENDANTS (default)**

Displays names from the specified logical name table and any descendant tables. (A descendant table is created by the CREATE/NAME\_TABLE command, with the PARENT\_TABLE qualifier specifying its parent table.) If you use the /DESCENDANTS qualifier, you must also use the /TABLE qualifier.

##### **/FULL**

Displays more detailed information on the access mode and any attributes for each logical name, equivalence string, and logical name table.



**/GROUP**  
**/JOB**  
**/PROCESS**  
**/SYSTEM**

Specifies the table from which the logical name is to be read. The **/GROUP** qualifier is synonymous with **/TABLE=LNМ\$GROUP**. The **/JOB** qualifier is synonymous with **/TABLE=LNМ\$JOB**. The **/PROCESS** qualifier is synonymous with **/TABLE=LNМ\$PROCESS**. The **/SYSTEM** qualifier is synonymous with **/TABLE=LNМ\$SYSTEM**.

**/JOB**  
See **/GROUP**.

**/OUTPUT[=file-spec]**  
**/NOOUTPUT**

Specifies the file to which the display is written; by default, the display is written to the current **SYS\$OUTPUT** device.

**/PROCESS**  
See **/GROUP**.

**/STRUCTURE**  
**/NOSTRUCTURE (default)**

Displays the family tree of all accessible tables. **/STRUCTURE** is mutually exclusive with all other qualifiers but **/OUTPUT**, **/ACCESS\_MODE**, and **/FULL**.

**/SYSTEM**  
See **/GROUP**.

**/TABLE=(name,...)**  
Requires **READ** access to display or search a shareable logical name table. Displays the specified logical name tables. Wildcards are allowed. Wildcarded names are used to match table names. Nonwildcarded names are treated both as table names and table search lists (whichever is appropriate).

## **SHOW MAGTAPE**

Displays the current characteristics and status of a specified magnetic tape device.

## **DCL-174    DCL Commands**

### **SHOW MAGTAPE**

#### **PARAMETER**

##### **device-name**

Specifies the name of the magnetic tape device for which you want to display the characteristics and status.

#### **QUALIFIER**

**/OUTPUT[=file-spec]**

**/NOOUTPUT**

Specifies the file to which the display is written; by default, the display is written to the current SYS\$OUTPUT device.

## **SHOW MEMORY**

Displays information about system resources related to memory.

#### **QUALIFIERS**

**/ALL (default)**

Displays information about paging and swapping files, physical memory, use of pool areas, process entry slots, and balance slots.

**/FILES**

Displays information about the paging and swap files.

**/FULL**

When used with the /FILES and /POOL qualifiers, displays additional information about the usage of each pool area or paging and swapping file.

**/OUTPUT[=file-spec]**

**/NOOUTPUT**

Specifies the file to which the display is written; by default, the display is written to the current SYS\$OUTPUT device.

**/PHYSICAL \_PAGES**

Displays information about the use of physical memory, including the number of pages in use and the number of pages on the free and modified page lists.

**/POOL**

Displays additional information about the use of fixed-size and dynamic pool areas, including the amount of used and free space and the size of the largest contiguous block for each pool.

**/SLOTS**

Displays information about the availability of process entry and balance slots.

## **SHOW NETWORK**

If your system is a routing node, displays the name and hardware line for each accessible network node, plus the number of logical links, the line cost, and the actual cost (hops) between your node and the other node. If your system is a nonrouting node, displays the designated router for your system.

### **QUALIFIERS**

**/OUTPUT[=file-spec]**

**/NOOUTPUT**

Specifies the file to which the display is written; by default, the display is written to the current SYS\$OUTPUT device.

## **SHOW PRINTER device-name**

Displays the characteristics of a printer.

### **PARAMETERS**

#### **device-name**

Name of the printer.

### **QUALIFIERS**

**/OUTPUT=[file-spec]**

Specifies the file to which the display is written; by default, the display is written to the current SYS\$OUTPUT device.

## **SHOW PROCESS [process-name]**

Displays information about your process and any current subprocesses. If no qualifier is entered, only a basic subset of information is displayed: the time, process terminal, user name and UIC, process name and process identification, priority, default directory, and allocated devices.

### **PARAMETERS**

#### **process-name**

Requires ownership of the process or that you have GROUP privilege and the process is in your group.

The name of the process about which information is to be displayed.  
Incompatible with the /IDENTIFICATION qualifier.

## **DCL-176    DCL Commands**

### **SHOW PROCESS**

#### **QUALIFIERS**

##### **/ACCOUNTING**

Displays the accumulated accounting statistics for the current session.

##### **/ALL**

Displays the basic subset of information plus the accounting statistics, privileges, quotas, and subprocesses.

##### **/CONTINUOUS**

Dynamically displays information about the specified process. Specify the process name as a parameter (process name defaults to the current process). Press the V key to display a map of the pages in the virtual address space of the process. Each character displayed in the map represents the type of page. If the current program counter (PC) is in the page, the page type is indicated by an at (@) sign. Pages locked in the working set are indicated by the letter L. Global pages are indicated by the letter G. Other valid pages in the working set are indicated by an asterisk. To terminate the display, press the E key. To return to the original display, press the spacebar.

##### **/IDENTIFICATION=pid**

Requires GROUP or WORLD privilege to access a process other than the current process.

Displays information about the process with the specified process identification. Incompatible with the process name parameter and the /SUBPROCESSES qualifier.

##### **/MEMORY**

Displays the process's use of dynamic memory areas.

##### **/OUTPUT[=file-spec]**

##### **/NOOUTPUT**

Specifies the file to which the display is written; by default, the display is written to the current SYS\$OUTPUT device.

##### **/PRIVILEGES**

Displays privileges and identifiers currently enabled for the process.

##### **/QUOTAS**

Displays, for each resource, either a quota or a limit. The values displayed for quotas reflect any quota reductions resulting from subprocess creation. The values displayed for limits reflect the resources available to a process at creation.

##### **/SUBPROCESSES**

Displays the current subprocesses in hierarchical order. Incompatible with the /IDENTIFICATION qualifier.

## SHOW PROTECTION

Displays your default file protection.

## SHOW QUEUE [queue-name]

Displays the current status of batch and print jobs.

### PARAMETERS

#### queue-name

Name of the queue in which the job exists. If *queue-name* is not specified, information on all queues is displayed. Wildcard characters are valid; the default queue name is \*.

### QUALIFIERS

#### /ALL

Displays all current and pending jobs in the specified queues.

#### /BATCH

Displays all batch queues and any jobs in those queues that are owned by the current process.

#### /BRIEF (default)

#### /FULL

Displays a brief or full description of the jobs in the queues.

#### /DEVICE

Displays jobs owned by the current process in all printer, terminal, and server queues.

#### /FILES

Requests a brief listing of information about job entries in the queue with the list of files associated with each job. The display includes a full file specification for each file in each job.

#### /FULL

See /BRIEF.

#### /OUTPUT[=file-spec]

#### /NOOUTPUT

Specifies the file to which the display is written; by default, the display is written to the current SYS\$OUTPUT device.

## **DCL-178    DCL Commands**

### **SHOW QUEUE/CHARACTERISTICS**

#### **SHOW QUEUE/CHARACTERISTICS [characteristic]**

Displays the names and numbers of available queue characteristics.

##### **PARAMETERS**

##### **characteristic**

The name of a characteristic. You can use wildcard characters. The default is \*.

##### **QUALIFIERS**

##### **/OUTPUT[=file-spec]**

##### **/NOOUTPUT**

Specifies the file to which the display is written; by default, the display is written to the current SYS\$OUTPUT device.

#### **SHOW QUEUE/FORM [form-name]**

Displays predefined form names and numbers that are available on queues.

##### **PARAMETERS**

##### **form-name**

The name of the form. You can use wildcard characters. The default is \*.

##### **QUALIFIERS**

##### **/BRIEF (default)**

##### **/FULL**

Displays a brief listing of information about the forms (form name, stock, number, and form description).

##### **/FULL**

See /BRIEF.

##### **/OUTPUT[=file-spec]**

##### **/NOOUTPUT**

Specifies the file to which the display is written; by default, the display is written to the current SYS\$OUTPUT device.

#### **SHOW QUOTA**

Requires READ access to the quota file in order to display the quotas of other users.

Displays the current disk quota that is authorized for 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.)



## QUALIFIERS

### **/DISK[=*device-name*[:]]**

Specifies the disk whose quotas are to be examined. By default, the current default disk is examined.

### **/USER=*uic***

Specifies which user's quotas are to be displayed. By default, the current user's quotas are displayed.

## **SHOW RMS\_DEFAULT**

Displays the current default multiblock count and multibuffer count that RMS uses for file operations.

## QUALIFIERS

### **/OUTPUT[=*file-spec*]**

### **/NOOUTPUT**

Specifies the file to which the display is written; by default, the display is written to the current SYS\$OUTPUT device.

## **SHOW STATUS**

Displays the current status of your process: the date, cumulative processor time used, cumulative buffered I/O operations performed, cumulative direct I/O operations performed, working set limit, amount of physical memory being used, number of open files, and cumulative page faults.

## **SHOW SYMBOL** [*symbol-name*]

Displays the value of the specified symbol.

## PARAMETERS

### ***symbol-name***

The name of the symbol. The *symbol-name* parameter is required if /ALL is not specified and is incompatible with /ALL. If /LOCAL or /GLOBAL is not specified, *symbol-name* means the first symbol at: (1) the current command level, (2) a higher command level, or (3) the global level, in that order. Wildcard characters are allowed in the *symbol-name* specification.

## **DCL-180    DCL Commands**

### **SHOW SYMBOL**

#### QUALIFIERS

##### **/ALL**

Displays the current values of all symbols in the specified symbol table.

##### **/GLOBAL**

##### **/LOCAL (default when /ALL is specified)**

Indicates the level of the symbol.

##### **/LOCAL**

See /GLOBAL.

##### **/LOG (default)**

##### **/NOLOG**

Generates a message indicating truncation of the symbol value. (The symbol value is truncated if it exceeds 255 characters.)

## **SHOW SYSTEM**

Displays status information concerning current processes: the time, process name and identification, processing state, priority, total process I/O, cumulative processor time used, cumulative page faults, amount of physical memory being used, and the type of process if not interactive (B for batch, N for network, S for subprocess).

#### QUALIFIERS

##### **/BATCH**

##### **/NETWORK**

##### **/PROCESS (default)**

##### **/SUBPROCESS**

Displays all batch jobs, all network processes, all processes, or all subprocesses on the system.

##### **/FULL**

Displays the default status information for current processes, plus process UICs.

##### **/OUTPUT[=file-spec]**

##### **/NOOUTPUT**

Specifies the file to which the display is written; by default, the display is written to the current SYS\$OUTPUT device.

## **SHOW TERMINAL [device-name]**

Displays the current characteristics of a terminal. (See SET TERMINAL for a list of the characteristics displayed.)

### **PARAMETERS**

#### **device-name**

Name of the terminal. The default is your terminal (SYS\$COMMAND).

### **QUALIFIERS**

#### **/OUTPUT[=file-spec]**

#### **/NOOUTPUT**

Specifies the file to which the display is written; by default, the display is written to the current SYS\$OUTPUT device.

#### **/PERMANENT**

Displays the permanent characteristics of the terminal. Use of the /PERMANENT qualifier requires LOG\_IO or PHY\_IO privilege.

## **SHOW [DAY]TIME**

Displays the current date and time.

## **SHOW TRANSLATION logical-name**

Searches one or more logical name tables for a specified logical name and returns the equivalence name of the first match found. If you do not specify the table, the tables specified by the multivalued logical name LNM\$DCL\_LOGICAL are searched. Unless LNM\$DCL\_LOGICAL has been redefined for your process, the process, job, group, and system logical name tables are searched in that order. The translation is not iterative.

### **PARAMETERS**

#### **logical-name**

The logical name to be displayed.

### **QUALIFIERS**

#### **/TABLE=name**

Searches the specified table. The default is /TABLE = LNM\$DCL\_LOGICAL.

## DCL-182    DCL Commands

### SHOW USERS

#### SHOW USERS [username]

Displays the user name, process name, terminal name, and process identification code (PID) of interactive users on the system.

#### PARAMETERS

##### **username**

The user about whom you want information. If you specify a string, all users whose user names begin with the string are displayed. If you omit *username*, a list of all interactive users is displayed.

#### QUALIFIERS

**/OUTPUT[=file-spec]**

**/NOOUTPUT**

Specifies the file to which the display is written; by default, the display is written to the current SYS\$OUTPUT device.

#### SHOW WORKING\_SET

Displays the working set limit, quota, and extent assigned to the current process.

#### QUALIFIERS

**/OUTPUT[=file-spec]**

**/NOOUTPUT**

Specifies the file to which the display is written; by default, the display is written to the current SYS\$OUTPUT device.

#### SORT input-file-spec,... output-file-spec

Sorts a file or files into one ordered output file.

#### PARAMETERS

##### **input-file-spec**

Specification of files whose records are to be sorted. All files must have the same record format and key description, but may have different file organizations. Wildcard characters are not allowed. The file type defaults to DAT.

##### **output-file-spec**

Specification of the file produced by the sort operation. Wildcard characters are not allowed. The file type defaults to the file type of the first input file.

## QUALIFIERS

### **/ALLOCATION=file-size**

Qualifies output-file-spec.

Required only to override relative and indexed-sequential input file characteristics. *File-size* is the number of 512-byte blocks to be allocated for the file and must be an integer in the range 1 through 4294967295. The /ALLOCATION qualifier is required if /CONTIGUOUS is specified.

### **/BUCKET\_SIZE=bucket-size**

Qualifies output-file-spec.

Required only to override input file characteristics. For relative and indexed files, specifies the bucket size in blocks. For input and output files of the same organization, the default is the same as the bucket size of the first input file; otherwise, the default is 1. Bucket size must be an integer in the range 1 through 32.

### **/COLLATING\_SEQUENCE=sequence**

Names the collating sequence for character data. *Sequence* can be ASCII, EBCDIC, or MULTINATIONAL. Note that when you specify EBCDIC, the characters remain in ASCII representation; only the order is changed. The /MULTINATIONAL qualifier specifies the collating sequence of the Multinational character set. See Appendix CHAR for the ASCII and Multinational character sets.

### **/CONTIGUOUS**

Qualifies output-file-spec.

Required only to override the first input file's characteristics. Specifies that the allocation of blocks for the output file be contiguous. /ALLOCATION must also be specified.

### **/DUPLICATES (default)**

### **/NODUPLICATES**

Deletes records with duplicate keys from the sort operation. The /NODUPLICATES qualifier is incompatible with the /STABLE qualifier.

### **/FORMAT=record-format[n:]**

Qualifies input-file-spec and output-file-spec.

Specifies record format and size. Possible options for the input file are:

RECORD\_SIZE=n      An integer in the range 1 through 32767 for sequential files, 1 through 16383 for relative files, or 1 through 16383 for indexed sequential files

FILE\_SIZE=n          An integer in the range 1 through 4294967295

## DCL-184    DCL Commands

### SORT

Possible options for the output file are:

FIXED=n	A fixed size record whose maximum size is an integer in the range 1 through 32767 for sequential files, 1 through 16383 for relative files, or 1 through 16383 for indexed sequential files
VARIABLE=n	A variable size record whose maximum size is an integer in the range 1 through 32767 for sequential files, 1 through 16383 for relative files, or 1 through 16383 for indexed sequential files
CONTROLLED=n	A controlled record whose size is an integer in the range 1 through 32767 for sequential files, 1 through 16383 for relative files, or 1 through 16383 for indexed sequential files
SIZE=n	An integer in the range 1 through 255

If /FORMAT is not specified for the output file, the format is based on the sort process selected: if RECORD or TAG sort is selected, the default is the format of the first input file; if ADDRESS or INDEX sort is selected, the default is FIXED.

#### /INDEXED\_SEQUENTIAL

#### /RELATIVE

#### /SEQUENTIAL

Qualifies output-file-spec.

Specifies the organization of the file. For a record or tag sort, the output file format defaults to the organization of the input file. For an indexed sort, the output file must exist and be empty and you must specify the /OVERLAY qualifier.

#### /KEY=(option,...)

Defines a sort key. You can specify /KEY up to 255 times to define 255 different key fields on which to sort. The default is a character data key, beginning in position 1 of the input record for a length of the LRL (longest record length) for the input file, up to a maximum length of 32767 bytes. The following keywords specify position, size, and data type of the key field within the record.



Option	Description
POSITION=start of key	Starting byte of the key within the record, where the first byte of the record is position 1. The value must be an integer in the range 1-32767. The position option is required.
CHARACTER (default)	Data type of the key.
BINARY	
F_FLOATING	
D_FLOATING	
G_FLOATING	
H_FLOATING	
ZONED	
DECIMAL	
PACKED_DECIMAL	
SIZE=n	Size of the key as follows depending on data type: <p>CHARACTER—Number of characters specified as an integer in the range 1-32767 (default = 32767)</p> <p>BINARY—Then integer value 1 (byte), 2 (word), 4 (longword), 8 (quadword), or 16</p> <p>DECIMAL—Number of digits, not counting the sign, specified as an integer in the range 1 to 31</p> <p>PACKED_DECIMAL—same as DECIMAL</p> <p>Do not specify a size for floating-point data types, whose sizes are fixed at 4 (F_FLOATING), 8 (D_ and G_FLOATING), and 16 (H_FLOATING) bytes. The total size of all keys must not exceed 32767 bytes.</p>
NUMBER=key order	Priority of the key specified as an integer in the range 1-255, where 1 means the primary key. The default is the order in which the keys are specified.
ASCENDING(default)	Order in which records are sorted for the key.
DESCENDING	

Option	Description
SIGNED (default)	Whether or not a sign is stored (binary keys only)
UNSIGNED	
TRAILING_SIGN (default)	Byte in which sign is stored — first or last (decimal keys only).
LEADING_SIGN	
OVERPUNCHED_SIGN (default)	Whether the sign is superimposed on the decimal value or is separated from the decimal (decimal keys only).
SEPARATE_SIGN	

**/OVERLAY**

**/NOOVERLAY**

Qualifies output-file-spec.

Writes the output to an existing file which must be empty. By default, a new output file is created for sequential and relative files. If the output file is INDEXED\_SEQUENTIAL, /OVERLAY must be specified.

**/PROCESS=keyword**

Defines the type of sort with one of the following keywords:

- |         |  |
|---------|--|
| RECORD  | Sorts complete records and produces an output file of complete records in sorted order (default).  |
| TAG     | Sorts keys, then reaccesses the input file to produce an output file of complete records in sorted order (the net result is the same as a record sort); terminal input is not allowed. |
| ADDRESS | Sorts keys and produces an output file of pointers; multiple input files are not allowed; terminal input is not allowed; the output file is in binary format.                          |
| INDEX   | Sorts keys and produces an output file of pointers and keys; multiple input files are not allowed; terminal input is not allowed; the output file is in binary format.                 |

**/RELATIVE**

See /INDEXED\_SEQUENTIAL.

**/SEQUENTIAL**

See /INDEXED\_SEQUENTIAL.

**/SPECIFICATION=file-spec**

Identifies the specification file to be used in the SORT operation. File type defaults to SRT. Any qualifiers specified in the SORT command line override the qualifiers in the specification file. The specification file can contain the following qualifiers:

## QUALIFIERS

`/CDD_PATH_NAME="cdd-path-name"`

Specifies a record definition ("*cdd-path-name*") from the Common Data Dictionary (CDD) if your system has VAX CDD installed. Once the fields have been identified, they may be used in later specification file qualifiers. (The `/CDD_PATH_NAME` qualifier may be used with or in place of the `/FIELD` qualifier.)

`/COLLATING_SEQUENCE=(SEQUENCE=sequence[,keyword=,...])`

Specifies the collating sequence for character key fields; the collating sequence can be ASCII (the default), EBCDIC, MULTINATIONAL, or a user-defined collating sequence that is specified as a string of characters (single or double) or a range of single characters. Each character and range must be separated by commas and enclosed in parentheses. You can also specify characters by their corresponding octal, decimal, or hexadecimal values, using the radix operators `%O`, `%D`, `%X`. Specify the quotation mark by doubling its occurrence within quotation marks by using a radix operator. Specify the null character with a radix operator (such as `%X0`). You must include in the sequence all characters that appear in the character keys or the character will be ignored (unless the `MODIFICATION` or `FOLD` keyword is specified).

Other optional keywords are as follows. Note that the `FOLD`, `MODIFICATION`, and `IGNORE` keywords are processed in the order in which they are specified.

### MODIFICATION

Specifies the change you want to make to the collating sequence (ASCII, EBCDIC, MULTINATIONAL, or user-defined). Use the format `MODIFICATION=(character operator character)`. Specify *character* as it is in the collating sequence and *operator* as `>`, `<`, or `=`. You can specify the following changes to a collating sequence:

- (1) Equate a character (single or double) to a character (single or double) that has already been assigned a collating value ("*a*"="*A*" or "*CH*"="*SH*" or "*C*"="*CH*").
- (2) Collate a character (single or double) after a single character that has already been assigned a collating value ("*CH*"`>`"*C*").
- (3) Collate a character (single or double) before a character that has already been assigned a collating value ("*CH*"`<`"*C*").

**SORT**

IGNORE	Specifies the character or range of characters to be initially ignored in the collating sequence (unless two or more strings have compared as equal and (1) TIE_BREAK is in effect or (2) the Multinational sequence is being used). Specify in the format IGNORE=character (or IGNORE=character range,...).
FOLD	Gives all lowercase letters the collating value of their uppercase equivalents (the Multinational sequence does this by default).
[NO]TIE_BREAK	Specifies whether or not numeric values are used to break any ties between characters that have equivalent values. (The Multinational sequence breaks ties in this way by default.)

/CONDITION=(NAME=condition-name,TEST=(field-name operator test[logical-operator,...]))

Defines a conditional test that can be used to change the relative order of a record (with the /KEY or /DATA qualifier) or to alter the contents of certain fields of a record (with the /OMIT or /INCLUDE qualifier). *Condition-name* specifies the name of the condition; once defined, you can use the condition name with the /KEY, /DATA, /OMIT, and /INCLUDE qualifiers. *Field-name* specifies the name of the field (defined by the /FIELD qualifier) being tested; *logical operator* specifies the logical (AND or OR) or relational (EQ, NE, GT, GE, LT, or LE) operator used in the test. *Operator-test* specifies the constant for which you are testing. Specify the constant with the following syntax: %D decimal\_digits, %O octal\_digits, %X hexadecimal\_digits, and "character".

/DATA=field-name

/DATA=(IF condition-name THEN "new-contents" ELSE "new-contents")

Specifies the fields to be directed to the output file and their order. (By default, the output file has the same record format as that of the input file.) Only the specified fields will appear in the output field. *Field-name* specifies the previously defined name of a field in a record; *condition-name* specifies a previously defined condition; and *new-contents* is either a constant or a field name that specifies how the record is to be altered.

/FIELD=(NAME=field-name,POSITION:n,SIZE:n, data-type) [DIGITS:n]

Defines the fields in the input files. (You must specify each field in the records to be merged, including key fields, field to be compared, and fields to be directed to your output file.) *Field-name* cannot have any embedded blanks, must begin with an alphabetic character, and can be no longer than 31 characters. POSITION specifies the position of the field in the record. SIZE specifies the size of a field, according to data type: character data must not exceed 32,767 characters; binary data must

be 1, 2, 4, 8, or 16 bytes; and floating-point data has no specified size. *Data-type* specifies the data type of the field; the default is character. Specify *data-type* as one of the following:

CHARACTER (default)	BINARY[,SIGNED]
BINARY,UNSIGNED	ZONED
D_FLOATING	F_FLOATING
G_FLOATING	H_FLOATING
PACKED_DECIMAL	DECIMAL,UNSIGNED
DECIMAL[,SIGNED,TRAILING_SIGN,OVERPUNCHED_SIGN]	
DECIMAL,LEADING_SIGN,SEPARATE_SIGN[,SIGNED]	
DECIMAL,LEADING_SIGN,[OVERPUNCHED_SIGN,SIGNED]	
DECIMAL,[TRAILING_SIGN],SEPARATE_SIGN[,SIGNED]	

DIGITS specifies the size of a field containing decimal data; *n* cannot exceed 31 digits.

**/INCLUDE=(CONDITION=condition-name,[KEY=...],DATA=...)**

Specifies that records are to be conditionally included (according to a previously defined condition). If you specify multiple **/INCLUDE** qualifiers, the order in which you specify them determines the order in which the input records are tested for inclusion. You unconditionally include any records not previously omitted or included by specifying **/INCLUDE** without a condition. The order of the key fields you specify determines how the internal key is built for merging; the order of the **DATA** fields determines the way in which the output record is formatted. If you specify a key or data field with **/INCLUDE**, you must define all other key or data fields in the record.

**/KEY={field-name[,order]}**

**/KEY=(IF condition-name THEN value ELSE value)**

Specifies key fields (up to 255) and the order of their priority (unnecessary if you are merging on the entire record using character data). *Field-name* is the name of the field specified in the **/FIELD** qualifier. *Order* can be **ASCENDING** or **DESCENDING**. The conditional form of the **/KEY** qualifier specifies a relative order of records; *value* can be a constant or a field name that has been defined in a **/FIELD** qualifier.

**/OMIT=(CONDITION=condition-name)**

Specifies records to be conditionally omitted from the output file (by a previously defined condition). If you specify multiple **/OMIT** qualifiers, the order in which you specify them determines the order in which the input records are tested for omission. You can unconditionally omit any records not previously omitted or included by specifying **/OMIT** without a condition.



## DCL-190    DCL Commands

### **SORT**

**/PAD=**single-character

Specifies a character to be used to fill an incomplete record when you are reformatting a record or comparing strings of unequal length; the null character is the default pad character. The pad character can be a character or a digit (decimal, octal, or hexadecimal). Enclose characters in quotation marks and precede digits with radix (%X23).

**/PROCESS=**type

Specifies the type of sort: type can be record (default), tag, address, or index. You cannot use address or index sort if the output records are going to be reformatted.

**/STABLE** (default)

**/NOSTABLE** (default)

Arranges records with equal keys in the output file in the order of the input files (by default, the order is unpredictable).

**/WORK\_FILES=**(disk,...)

Assigns work files to the specified disk and/or diskette. (By default, work files are located in the directory SYS\$SCRATCH; placing them on separate disks with /WORK\_FILES permits overlap of SORT's read/write cycle.)

**/STABLE**

**/NOSTABLE** (default)

Maintains the order of records with identical keys; otherwise, the order is unpredictable. The /STABLE qualifier is incompatible with the /NODUPPLICATES qualifier.

**/STATISTICS**

**/NOSTATISTICS** (default)

Displays a statistical summary at the end of the sort.

**/WORK\_FILES=**number-of-files

Specifies the number of temporary work files. The value can be from 0 through 10, with a default of 2.

**SPAWN** [command-string]

Requires TMPMBX or PRMMBX user privilege.

Creates a subprocess.



## PARAMETERS

### **command-string**

A command string of less than 132 characters that is to be executed in the context of the created subprocess. When the command completes execution, the subprocess terminates and control returns to the parent process. If both a command string and the /INPUT qualifier are specified, the specified command string executes before additional commands are obtained from the /INPUT qualifier.

## QUALIFIERS

### **/CARRIAGE\_CONTROL**

### **/NOCARRIAGE\_CONTROL**

Prefixes carriage return/line feed characters to the subprocess's prompt string. The default is the parent process's setting.

### **/CLI=cli-file-spec**

### **/NOCLI**

Specifies the command language interpreter (CLI) that will be used by the subprocess. The default CLI is that defined in SYSUAF. If you specify /CLI, context is copied to the subprocess.

### **/INPUT=file-spec**

Specifies an input file containing one or more DCL commands to be executed by the subprocess. File type defaults to COM. Once processing of the input file is complete, the subprocess is terminated. If both a command string and the /INPUT qualifier are specified, the specified command string executes before additional commands are obtained from the /INPUT qualifier. If none is specified, SYS\$INPUT is assumed (in which case a SPAWN/NOWAIT will be aborted if CTRL/Y is typed to abort something running in your parent process).

### **/KEYPAD (default)**

### **/NOKEYPAD**

Copies keypad key definitions and the current keypad state from the parent process.

### **/LOG (default)**

### **/NOLOG**

Displays the assigned subprocess name and any messages indicating transfer of control between processes.

### **/LOGICAL\_NAMES (default)**

### **/NOLOGICAL\_NAMES**

Copies process logical names and logical name tables (except those explicitly marked CONFINE or created in executive or kernel mode) to the subprocess.

## **DCL-192    DCL Commands**

### **SPAWN**

#### **/NOTIFY**

#### **/NONOTIFY (default)**

Broadcasts a message to your terminal notifying you that your subprocess has completed or aborted. Incompatible with the /NOWAIT qualifier or the execution of SPAWN from within a noninteractive process.

#### **/OUTPUT=file-spec**

Specifies the output file to which the results of the SPAWN operation are written. (Do not specify SYS\$COMMAND as a file specification with the /NOWAIT qualifier; both parent and subprocess output will be displayed simultaneously on your terminal.)

#### **/PROCESS=subprocess-name**

Specifies the name of the subprocess to be created. The default subprocess name format is: username\_n.

#### **/PROMPT[=string]**

Specifies the prompt string for DCL to use in the subprocess. The default is the prompt of the parent process. The string must be enclosed in quotation marks if it contains spaces, special characters, or lowercase characters.

#### **/SYMBOLS (default)**

#### **/NOSYMBOLS**

Passes global and local symbols (except \$RESTART, \$SEVERITY, and \$STATUS) to the subprocess.

#### **/TABLE=command=table**

Specifies the name of an alternate command table to be used by the subprocess.

#### **/WAIT (default)**

#### **/NOWAIT**

Requires that you wait for the subprocess to terminate before you issue another DCL command. The /NOWAIT qualifier allows you to issue new commands while the subprocess is running. (Use the /OUTPUT qualifier with the /NOWAIT qualifier to avoid displaying both parent and subprocess output on the terminal simultaneously.)

### **START/QUEUE queue-name**

Requires OPER privilege or EXECUTE access to the queue.

Starts or restarts the specified queue after it has been initialized. The /TOP\_OF\_FILE, /BACKWARD or /FORWARD, /SEARCH, and /ALIGN qualifiers are processed in that order when more than one occurs in a command line.

## PARAMETERS

### **queue-name**

Name of the queue.

## QUALIFIERS

### **/ALIGN[=(option,...)]**

Prints alignment pages that enable the operator to properly align the forms in the printer or terminal. Use this qualifier in restarting an output queue from a paused state. Possible options are:

- |      |  |
|------|--|
| MASK | Displays alphabetic characters as x's and numbers as 9's; nonalphanumeric characters are not masked. The default is not to mask. |
| n    | Specifies the number of alignment pages to print. The value of <i>n</i> can be from 1 to 20; the default is 1.                   |

### **/BACKWARD=n**

Restarts a print queue *n* pages before the current page; *n* defaults to 1. Use this qualifier in restarting an output queue from a paused state.

### **/BASE\_PRIORITY=n**

Specifies the process priority base at which jobs are initiated from a batch queue. The value of *n* can be from 0 through 15. By default, jobs are initiated at the priority established by DEFPRI at system generation (usually 4).

### **/BATCH**

#### **/NOBATCH (default)**

Specifies that the queue is a batch queue. (The queue must have been initialized as a batch queue.)

### **/BLOCK\_LIMIT=([lower,]upper)**

#### **/NOBLOCK\_LIMIT (default)**

Restricts the size of print jobs that can be executed on a printer or terminal queue. The lower parameter specifies the minimum number of blocks that will be accepted by the queue for a print job. The upper parameter specifies the maximum number of blocks that will be accepted by the queue for a print job. If a job contains fewer blocks than the number specified by the lower parameter or more blocks than the number specified by the upper parameter, the job remains pending until the block limit for the queue is changed. To specify only the lower parameter, you must use two sets of quotation marks (""") in place of the upper specifier.

### **/CHARACTERISTICS=(characteristic,...)**

#### **/NOCHARACTERISTICS (default)**

Specifies one or more characteristics for processing jobs on the queue. Each time you specify **/CHARACTERISTIC**, all previously set characteristics are erased. A

## DCL-194    DCL Commands

### START/QUEUE

queue must have all the characteristics specified for the job or the job remains pending.

#### **/CPUDEFAULT=time**

Specifies the default CPU time limit for batch jobs. Time can be specified as delta time, 0, NONE, or INFINITE. Both the value 0 and the keyword INFINITE allow unlimited CPU time (subject to the restrictions imposed by the /CPUMAXIMUM qualifier or the user authorization file); the keyword NONE indicates that no time limit is needed.

#### **/CPUMAXIMUM=time**

Specifies the maximum CPU time limit for batch jobs. The /CPUMAXIMUM qualifier overrides the time limit specified in the user authorization file (UAF). Time can be specified as delta time, 0, NONE, or INFINITE. Both the value 0 and the keyword INFINITE allow unlimited CPU time; the keyword NONE specifies that no time limit is needed.

#### **/DEFAULT=(option,...)**

##### **/NODEFAULT**

Establishes default options for the PRINT command. The /DEFAULT qualifier can not be used with the /GENERIC qualifier. Possible options are:

[NO]BURST[=keyword]	Specifies where to print burst pages (flag pages that are printed over the paper's perforations for easy identification of individual files in a print job). The keyword ALL (the default) places burst pages before each printed file in the job. The keyword ONE places a burst page before the first printed file in the job.
[NO]FEED	Specifies whether a form-feed is automatically inserted at the end of a page
[NO]FLAG[=keyword]	Specifies where to print flag pages (containing the job entry number, the name of the user submitting the job, and so on). The keyword ALL places flag pages before each printed file in the job. The keyword ONE places a flag page before the first printed file in the job.

**FORM=type**

Specifies the default form for a printer, terminal, or server queue. If a job is not submitted with an explicit form definition, then this form will be used to process the job. The systemwide default form, form=0, is the default value for this keyword. See also /FORM\_MOUNTED.

**[NO]TRAILER[=keyword]**

Specifies where to print trailer pages. The keyword ALL places trailer pages after each printed file in the job. The keyword ONE places a trailer page after the last printed file in the job.

If you specify any of the keywords BURST, FLAG, TRAILER without specifying a value, the value ALL is used by default.

**/DISABLE\_SWAPPING**

**/NODISABLE\_SWAPPING (default)**

Controls whether batch jobs executed from a queue can be swapped in and out of memory.

**/ENABLE\_GENERIC (default)**

**/NOENABLE\_GENERIC**

Allows files queued to a generic queue that does not specify explicit queue names in the /GENERIC qualifier to be placed in this execution queue for processing.

**/FORM\_MOUNTED=type**

Specifies the form type for a printer, terminal, or server queue. If the stock of the mounted form is not identical to the stock of the default form, as indicated by the DCL command qualifier /DEFAULT=FORM=type, then all jobs submitted to this queue without an explicit form definition will enter a pending state. If a job is submitted with an explicit form and the stock of the explicit form is not identical to the stock of the mounted form, then the job will enter a pending state. In both cases, the pending state will be maintained until the stock of the mounted form of the queue is identical to the stock of the form associated with the job. The /FORM\_MOUNTED qualifier can not be used with the /GENERIC qualifier.

**/FORWARD=n**

Advances the specified number of pages before resuming printing the current file; the default is 1. Use this qualifier in restarting an output queue from a paused state.

**/GENERIC[=(queue-name,...)]**

**/NOGENERIC (default)**

Specifies that the queue is generic; jobs in a generic queue are moved to one of the specified queues for processing. (The /BATCH qualifier of the generic



## **DCL-196     DCL Commands**

### **START/QUEUE**

queue and all the specified execution queues must match.) If you do not specify a queue, jobs can move to any execution queue (initialized without the printer, terminal, or server) as the generic queue. (For a generic server queue, the /PROCESSOR qualifier of the generic queue and the execution queue must also match.) By default, a generic queue is a print queue; use the appropriate qualifier (/BATCH, /PROCESSOR, /TERMINAL) to override the default. The /GENERIC qualifier is incompatible with the /DEFAULT, /FORM\_MOUNTED, and /SEPARATE qualifiers.

#### **/JOB\_LIMIT=n**

Specifies the number of batch jobs that can be executed concurrently from the queue; the value of *n* defaults to 1.

#### **/LIBRARY=filename**

Specifies the file name for the device control library. (The /LIBRARY qualifier can be used to specify an alternate device control library when used to initialize a symbiont queue.) The default library is SYS\$LIBRARY:SYSDEVCTL.TLB. You can specify only a file name; the library must be in SYS\$LIBRARY and the file type must be TLB.

#### **/NEXT**

Restarts the queue with the next job. (By default, the job that was executing when the queue stopped resumes if it has not been deleted.)

#### **/ON=device[:]**

Specifies the device on which this execution queue is located; the default device name is the same as the queue name. For batch queues, you can only specify the node name.

#### **/OWNER\_UIC=uic**

Requires OPER privilege.

Specifies a UIC for the queue. The default UIC is [1,4].

#### **/PROCESSOR=filename**

#### **/NOPROCESSOR**

Used for a symbiont queue, specifies a print symbiont image in SYS\$SYSTEM:filename.EXE; the default file name is PRTSMB. Used for a generic queue, the /PROCESSOR qualifier specifies that the generic queue can place jobs only on queues that have declared themselves as server queues and that are executing the specified symbiont image.

#### **/PROTECTION=(ownership[:access],...)**

Requires OPER privilege.

Applies the specified protection to the queue. The ownership categories are SYSTEM, OWNER, GROUP, WORLD; the access categories are R (read),



W (write), E (create), and D (delete). The default protection is (SYSTEM:E, OWNER:D, GROUP:R, WORLD:W).

**/RECORD\_BLOCKING (default)**

**/NORECORD\_BLOCKING**

Determines whether the symbiont can concatenate (or block together) output records for transmission to the output device. If you specify /NORECORD\_BLOCKING, the symbiont is directed to send each formatted record in a separate I/O request to the output device. For the standard MicroVMS print symbiont, record blocking can have a significant performance advantage over single-record mode.

**/RETAIN[=keyword]**

**/NORETAIN (default)**

Retains jobs in the queue in a completed status after they have executed. Possible keywords are:

ALL	Retains all jobs in the queue after execution (default)
ERROR	Retains in the queue only jobs that complete unsuccessfully

**/SCHEDULE=SIZE (default)**

**/SCHEDULE=NOSIZE**

Specifies whether pending jobs in a printer or terminal queue are scheduled for printing based on the size of the job. When the default, /SCHEDULE=SIZE, is in effect, shorter jobs will print before longer ones.

Note: If you issue this command while there are pending jobs in any queue, the effect on future jobs is unpredictable.

**/SEARCH="string"**

Resumes printing the current file on the first page containing the specified string. The string can be from 1 through 63 characters and must be enclosed in quotation marks. Use this qualifier in restarting an output queue from a paused state.

**/SEPARATE=(keyword,...)**

**/NOSEPARATE (default)**

Specifies the job separation defaults for a printer or terminal queue. The /SEPARATE qualifier can not be used with the /GENERIC qualifier. Possible keywords are:

[NO]BURST	Prints a burst page (a flag page printed over the paper's perforations for easy identification of individual files) at the beginning of every job.
-----------	--

## **DCL-198    DCL Commands**

### **START/QUEUE**

[NO]FLAG	Prints a flag page (containing the job entry number, the name of the user submitting the job, and so on) at the beginning of every job.
[NO]TRAILER	Prints a trailer page at the end of every job.
[NO]RESET=(m,...)	Specifies a job reset sequence for the queue. The specified modules from the device control library (see /LIBRARY) are used to reset the device each time a job reset occurs.

#### **/TERMINAL**

#### **/NOTERMINAL (default)**

Associates a generic queue with terminal queues (instead of printer queues) with matching characteristics.

#### **/TOP\_OF\_FILE**

Resumes printing at the beginning of the file that was current when the queue paused. Use this qualifier only when restarting an output queue from a paused state.

#### **/WSDEFAULT=n**

Defines a working set default for a batch job. The /WSDEFAULT qualifier overrides the working set size specified in the user authorization file. Possible values are: a positive integer in the range 1 through 65,535, 0, or the keyword NONE (the default). A zero or NONE sets the default value to the value specified either in the UAF or by the SUBMIT command (if specified).

When used for an output queue, this qualifier specifies the working set default of a symbiont process for a printer, terminal, or server queue when the symbiont process is created.

#### **/WSEXTENT=n**

Defines a working set extent for the batch job. The /WSEXTENT qualifier overrides the working set extent in the user authorization file. Possible values are: a positive integer in the range 1 through 65,535, 0, or the keyword NONE (the default). A zero or NONE sets the default value to the value specified either in the UAF or by the SUBMIT command (if specified).

When used for an output queue, this qualifier specifies the working set extent of a symbiont process for a printer, terminal, or server queue when the symbiont process is created.

#### **/WSQUOTA=n**

Defines a working set page size (working set quota) for the batch job. The /WSQUOTA qualifier overrides the value in the user authorization file. Possible values are: a positive integer in the range 1 through 65,535, 0, or the keyword

NONE (the default). A zero or NONE sets the default value to the value specified either in the UAF or by the SUBMIT command (if specified).

When used for an output queue, this qualifier specifies the working set quota of a symbiont process for a printer, terminal, or server queue when the symbiont process is created.

## **START/QUEUE/MANAGER [file-spec]**

Requires both OPER and SYSNAM privilege.

Starts the queue manager for the batch/print facility and opens the job queue manager file. The START/QUEUE/MANAGER command must be executed before you can execute any other queue management or job submission command.

### **PARAMETERS**

#### **file-spec**

The name of the file containing the information about batch and print jobs, queues, and form definitions. The default file specification is SYS\$SYSTEM:JBCSYSQUE.DAT.

### **QUALIFIERS**

#### **/BUFFER\_COUNT=n**

Specifies the number of buffers in a local buffer cache to allocate for performing I/O operations to the system job queue file. Specify *n* as a positive integer in the range of 1 through 127 or 0. If 0 is specified, the default value of *n*=50 is used.

#### **/EXTEND\_QUANTITY=n**

Specifies the number of blocks by which the system job queue file is extended (when this action is necessary). This value is also used as the initial allocation size when the queue file is created. Specify *n* as a positive integer in the range of 10 through 65,535 or 0. If 0 is specified, the default value of *n*=100 is used.

#### **/NEW\_VERSION**

#### **/NONEW\_VERSION (default)**

Specifies that a new version of the job queue manager file be created to supersede an existing version. All jobs in the previous version are lost if a new version is specified.

#### **/RESTART**

#### **/NORESTART (default)**

The /RESTART qualifier specifies that the queue manager be restarted automatically on recovery from a job controller abort. In addition, batch and output queues are restored to the states that existed prior to the interruption of service. The job queue manager file that is opened is the same file that was open before the abort. Upon restarting, the job controller uses the default values for

## **DCL-200    DCL Commands**

### **START/QUEUE/MANAGER**

the /EXTEND\_QUANTITY and /BUFFER\_COUNT qualifiers. Previously set values are lost.

When the job controller incurs an internal fatal error, the process aborts and restarts itself. By default, the queue manager is not restarted. Intervention by a user with OPERATOR privilege is necessary to restart the queue manager and to restore the queueing environment using START/QUEUE/MANAGER and appropriate START/QUEUE commands.

Note: In order to prevent a looping condition, the job controller will not restart the queue manager if it detects an error within two minutes of starting the queue manager.

### **STOP [process-name]**

Requires GROUP privilege to stop other processes in the same group.

Terminates execution of a command, image, a command procedure, a command procedure that was interrupted by CTRL/Y, or a detached process or subprocess.

#### **PARAMETERS**

##### **process-name**

Requires that the process be in your group.

Name of the process running the command procedure or image. The process name can have from 1 to 15 alphanumeric characters. Incompatible with the /IDENTIFICATION qualifier. You must use /IDENTIFICATION=pid to specify a process outside of your group.

#### **QUALIFIERS**

##### **/IDENTIFICATION=pid**

Specifies the system-assigned process identification. /IDENTIFICATION can be used in place of the process name parameter.

### **STOP/QUEUE queue-name**

Requires OPER (operator) privilege or EXECUTE access to the queue.

Stops the specified execution queue. All jobs currently executing in the queue are suspended (until the queue is restarted with the START/QUEUE command) and no new jobs can be initiated. The /REQUEUE and /RESET qualifiers provide other ways of stopping queues. To stop individual jobs on the queue, specify the /ABORT, /ENTRY, or /REQUEUE qualifier.

## PARAMETERS

### **queue-name**

Name of the queue.

## QUALIFIERS

### **/ABORT**

Aborts the current print job, deleting it from the queue, and resumes execution of the jobs in the queue. (STOP/QUEUE/ABORT is equivalent to STOP/ABORT.)

### **/ENTRY=job-number**

Stops the currently executing job on the specified batch queue.

### **/HOLD**

Places the aborted job on hold; must be used in combination with the /REQUEUE qualifier. To release the job, use the SET QUEUE/ENTRY /RELEASE or SET QUEUE/ENTRY/NOHOLD command; to delete, specify DELETE/ENTRY.

### **/NEXT queue-name**

Stops the queue after all executing jobs have completed processing. No new jobs can be initiated; the START/QUEUE command will restart the queue.

### **/PRIORITY=n**

Requires OPER or ALTPRI privilege to raise the priority above the value of the MAXQUEPRI parameter.

Changes the priority of the aborted job. Must be used with the /REQUEUE qualifier. The value of *n* is an integer from 0 to 255 that specifies priority; the default is the current priority of the job.

### **/REQUEUE[=queue-name]**

Stops the current job and requeues it for later processing. Print jobs that have been checkpointed will resume printing at the checkpoint. Batch jobs containing SET RESTART\_VALUE commands will run those portions of the job that have not successfully completed. When you use /REQUEUE with a batch queue, you must also use /ENTRY. If you specify a queue name, the current job is transferred to another queue.

### **/RESET**

Stops the queue without first terminating currently executing jobs. The START /QUEUE command restarts the queue. Current jobs that are restartable (all print jobs and any batch jobs submitted with the /RESTART qualifier) will be requeued for processing. Current jobs that are not restartable are aborted and must be resubmitted for processing.



**DCL-202     DCL Commands**  
**STOP/ABORT**

**STOP/ABORT queue-name[:]**

Requires OPER privilege, EXECUTE access to the queue, or DELETE access to the current job.

Aborts the executing print job, deleting it from the queue, and resumes execution of jobs in the queue. (The STOP/ABORT command is equivalent to the STOP/QUEUE/ABORT command.)

**PARAMETERS**

**queue-name**

The name of the queue in which the job is executing.

**STOP/ENTRY =entry-number queue-name[:]**

Requires OPER (operator) privilege, EXECUTE access to the queue, or DELETE access to the current job.

Stops the specified job currently executing in the specified batch queue and resumes execution of the next pending job in the queue. The job number is the number assigned to the job when it is submitted to the queue. (The STOP/ENTRY command is equivalent to the STOP/QUEUE/ENTRY command; use the DELETE/ENTRY command to stop an entry that is queued and awaiting execution.)

**PARAMETERS**

**queue-name**

The name of the batch queue in which the job is executing.

**STOP/QUEUE/MANAGER**

Requires both OPER and SYSNAM privilege.

Performs an orderly shutdown of the system job queue manager.

**STOP/REQUEUE [=queue-name] queue-name[:]**

Requires OPER (operator) privilege, EXECUTE access to the queue or DELETE access to the current job.

Stops the current job on the specified queue and requeues it for later processing; execution of the next pending job in the queue resumes. Print jobs that have been checkpointed will resume printing at the checkpoint. Batch jobs (specify /ENTRY) containing SET RESTART\_VALUE commands will run those portions of the job that have not successfully completed.



## PARAMETERS

### **queue-name**

The first *queue-name* ([=queue-name]) optionally specifies a queue to which the job is to be requeued. The second, required *queue-name* (queue-name[:]) specifies the name of the queue in which the job is executing.

## QUALIFIERS

### **/ENTRY=job-number**

Required with batch queues to specify the job; the *job-number* is the number assigned to the job when it was submitted to the queue.

### **/HOLD**

Places the aborted job in a hold state for later release with the SET/QUEUE /ENTRY/RELEASE or SET QUEUE/ENTRY/NOHOLD command. (Use DELETE/ENTRY to delete a job in the hold state.)

### **/PRIORITY=n**

Requires either OPER privilege or ALTPRI privilege to raise the priority value above the value of the SYSGEN parameter MAXQUEPRI.

Changes the priority of the requeued job. The value of *n* can be from 0 to 255; the default value of *n* is the job's current priority value.

## **SUBMIT file-spec,...**

Requires OPER privilege, E (execute) access to the queue, or W (write) access to the queue.

Queues a batch job.

## PARAMETERS

### **file-spec**

Name of a file containing a command procedure. Wildcard characters are allowed. The file type defaults to COM. If a node name is specified, the /REMOTE qualifier must also be specified.

## QUALIFIERS

### **/AFTER=absolute-time**

Holds the job until the specified time. If the time has passed, processes the job immediately. Time can be an absolute time or a combination of absolute and delta times.

**DCL-204     DCL Commands**  
**SUBMIT**

**/BACKUP**  
**/NOBACKUP**

Selects files according to the dates of their most recent backups. Relevant only with the /BEFORE and /SINCE qualifiers.

**/BEFORE[=time]**  
**/NOBEFORE**

Submits only those files dated before the specified time. You can specify time as an absolute time, as a combination of absolute and delta times, or as one of the following keywords: TODAY (default), TOMORROW, or YESTERDAY. Specified with /BACKUP, /CREATED (default), /EXPIRED, or /MODIFIED.

**/BY\_OWNER[=uic]**

Submits only those files with the specified user identification code. The default UIC is that of the current process.

**/CHARACTERISTICS=(characteristic,...)**

Specifies one or more characteristics for processing the job. A job must have all the characteristics specified for the job (or it remains pending).

**/CLI=file-spec**

Specifies the command language interpreter (CLI) that will be used in processing the job. The file name specifies that the CLI be SYS\$SYSTEM:filename.EXE. The default CLI is that defined in the user authorization file.

**/CONFIRM**  
**/NOCONFIRM (default)**

Requests confirmation before submitting each file. The following responses are valid:

YES	Submit the file
NO	Do not submit the file
TRUE	Submit the file
FALSE	Do not submit the file
1	Submit the file
0	Do not submit the file
RETURN	Do not submit the file
ALL	Continue execution of the command with no further confirmation prompts

CTRL/Z     Stop execution of the command  
QUIT     Stop execution of the command

**/CPUTIME=keyword**

Specifies a CPU time limit for the batch job. Time can be specified as delta time, 0, NONE, or INFINITE. Both the value 0 and the keyword INFINITE allow unlimited CPU time; the keyword NONE defaults to your user authorization file (UAF) value or the limit specified on the queue. Note that you cannot specify more time than permitted by the base queue limits or your own UAF.

**/CREATED**

**/NOCREATED**

Selects files based on their dates of creation. Relevant only with the /BEFORE and /SINCE qualifiers.

**/DELETE**

**/NODELETE (default)**

Positional qualifier.

If you specify the /DELETE qualifier after the SUBMIT command name, all files are deleted after executing them. If you specify the /DELETE qualifier after a file specification, only that file is deleted after it is executed.

**/EXCLUDE=(file-spec,...)**

**/NOEXCLUDE**

Excludes files from the SUBMIT operation. The qualifier value file-spec cannot include a device name. Wildcard characters are supported for file specifications. However, you cannot use relative version numbers to exclude a specific version.

**/EXPIRED**

**/NOEXPIRED**

Selects files based on their dates of expiration. Relevant only with the /BEFORE and /SINCE qualifiers.

**/HOLD**

**/NOHOLD (default)**

Holds the job (until released by the SET QUEUE/ENTRY command).

**/IDENTIFY (default)**

**/NOIDENTIFY**

Displays the queue name and job number of the job when it is queued.

**DCL-206     DCL Commands**  
**SUBMIT**

**/KEEP**  
**/NOKEEP**

Saves the log file after printing it; /NOKEEP is the default unless /NOPRINTER is specified.

**/LOG\_FILE=file-spec**  
**/NOLOG\_FILE**

Names the log file. The default is job-name.LOG. You can use /LOG\_FILE to specify a different device. Logical names in the file specification are translated in the context of the process that submits the job.

**/MODIFIED**  
**/NOMODIFIED**

Selects files according to the dates on which they were last modified. Relevant only with the /BEFORE and /SINCE qualifiers.

**/NAME=job-name**

Names the job (and possibly the batch job log file). The name must be 1 through 39 alphanumeric characters. Enclose the name in parentheses if it contains any special characters. The default is the name of the first file in the job.

**/NOTIFY**  
**/NONOTIFY (default)**

Broadcasts a message to your terminal when the job completes.

**/PARAMETERS=(parameter,...)**

Provides the values of up to 8 optional parameters (equated to P1 through P8, respectively, in each command procedure in the job). The symbols are local to the specified command procedure. If the parameter contains spaces, special characters, or lowercase characters, enclose it in quotation marks. The size of the parameter (including enclosing quotation marks and the preceding comma) must not exceed 255 characters. However, the total length of all eight parameter strings of the /PARAMETER qualifier cannot exceed 480 characters.

**/PRINTER=queue-name**  
**/NOPRINTER**

Queues the log file to the specified queue for printing. The default is SYS\$PRINT.

**/PRIORITY=n**

Requires OPER or ALTPRI privilege to raise the priority above the value of the MAXQUEPRI parameter.

Specifies the job-scheduling priority for the specified job. The value of n is an integer from 0 to 255; the default is the value of the SYSGEN parameter DEFQUEPRI.

**/QUEUE=queue-name**

Queues the job to the specified batch queue. The default is SYS\$BATCH.

**/REMOTE**

Queues the job to a queue on another node. The node name must be included in the file specification. No other qualifiers can be specified with /REMOTE.

**/RESTART**

**/NORESTART (default)**

Restarts the job after a system failure or a STOP/REQUEUE command.

**/SINCE[=time]**

**/NOSINCE**

Submits only those files dated after the specified time. You can specify time as an absolute time, as a combination of absolute and delta times, or as one of the following keywords: TODAY (default), TOMORROW, or YESTERDAY. Specified with /BACKUP, /CREATED (default), /EXPIRED, or /MODIFIED.

**/USER=username**

Requires CMKRNL privilege and R (read) access to the user authorization file (UAF).

Submits a job for another user so that the job runs under the *username* and UIC of the other user.

**/WSDEFAULT=n**

Defines a working set default for a batch job; the /WSDEFAULT qualifier overrides the working set size specified in the user authorization file. Possible values of *n* are a positive integer in the range 1 through 65,535, 0, or the keyword NONE. A value of 0 or the keyword NONE sets the default value to the value specified either in the UAF or by the working set quota established for the queue. You cannot request a value higher than the default.

**/WSEXTENT=n**

Defines a working set extent for the batch job; the /WSEXTENT qualifier overrides the working set extent in the user authorization file. Possible values of *n* are a positive integer in the range 1 through 65,535, 0, or the keyword NONE. A value of 0 or the keyword NONE sets the default value either to the value specified in the UAF or set for the queue. You cannot request a value higher than the default.

**/WSQUOTA=n**

Defines a working set page size (working set quota) for the batch job; the /WSQUOTA qualifier overrides the value in the user authorization file. Possible values of *n* are a positive integer in the range 1 through 65,535, 0, or the keyword NONE. A value of 0 or the keyword NONE sets the default value to

## **DCL-208     DCL Commands**

### **SUBMIT**

the value specified either in the UAF or set for the queue. You cannot request a value higher than the default.

### **SYNCHRONIZE [job-name]**

Holds your process until the specified batch job terminates.

#### **PARAMETERS**

##### **job-name**

Name of the job (as specified in the SUBMIT command). If you have more than one job with the same name, the synchronization occurs for the last job submitted. The job name parameter is overridden by the /ENTRY qualifier.

#### **QUALIFIERS**

##### **/ENTRY=job-number**

Identifies the job by the system assigned job number. The /ENTRY qualifier overrides any job name specified. You must specify either the job name or the /ENTRY qualifier.

##### **/QUEUE=queue-name**

Names the queue containing the job. The default is SYS\$BATCH.

### **TYPE file-spec,...**

Displays a file or files in text format on SYS\$OUTPUT (your terminal unless you reassign SYS\$OUTPUT).

#### **PARAMETERS**

##### **file-spec**

The specification of the file being displayed. Wildcard characters are allowed. The plus sign can be used in place of the comma between file specifications. The file type defaults to LIS.

#### **QUALIFIERS**

##### **/BACKUP**

##### **/CREATED (default)**

##### **/EXPIRED**

##### **/MODIFIED**

Selects files according to the dates of their most recent backups, their creation dates, their expiration dates, or the dates of their last modifications. Relevant only with the /BEFORE and /SINCE qualifiers.



**/BEFORE[=time]**

Displays only those files with dates that precede the specified time. You can specify time as absolute or a combination of absolute and delta times, or as one of the following keywords: TODAY (default), TOMORROW, or YESTERDAY. Specified with /BACKUP, /CREATED (default), /EXPIRED, or /MODIFIED.

**/BY\_OWNER[=uic]**

Displays only those files with the specified user identification code. The default UIC is that of the current process.

**/CONFIRM**

**/NOCONFIRM (default)**

Requests confirmation before displaying each file. The following responses are valid:

YES	Type the file
NO	Do not type the file
TRUE	Type the file
FALSE	Do not type the file
1	Type the file
0	Do not type the file
RETURN	Do not type the file
ALL	Continue execution of the command with no further confirmation prompts
CTRL/Z	Stop execution of the command
QUIT	Stop execution of the command

**/CREATED (default)**

See /BACKUP.

**/EXCLUDE=(file-spec,...)**

Excludes the specified files from display. The qualifier value *file-spec* cannot include a device name. Wildcard characters are supported for file specifications. However, you cannot use relative version numbers to exclude a specific version.

**/EXPIRED**

See /BACKUP.

**/MODIFIED**

See /BACKUP.

## **DCL-210    DCL Commands**

### **TYPE**

**/OUTPUT[=file-spec]**  
**/NOOUTPUT**

Names a file, rather than SYS\$OUTPUT, to receive the typed output.

**/PAGE**  
**/NOPAGE (default)**

Displays a screenful of the specified file with each RETURN.

**/SINCE[=time]**

Selects for display only those files dated after the specified time. You can specify time as absolute or a combination of absolute and delta times, or as one of the following keywords: TODAY (default), TOMORROW, or YESTERDAY. Specified with /BACKUP, /CREATED (default), /EXPIRED, or /MODIFIED.

## **UNLOCK file-spec,...**

Makes an improperly closed file accessible.

### **PARAMETERS**

#### **file-spec**

Specification of file to be unlocked. Wildcard characters are allowed. The plus sign can be used in place of the comma between file names.

### **QUALIFIERS**

**/CONFIRM**  
**/NOCONFIRM (default)**

For each file being unlocked, displays a query to which you must respond Y or T to unlock the file. Any other response aborts the unlock operation.

**/LOG**  
**/NOLOG (default)**

Displays the file specification of each file being unlocked.

## **WAIT time**

Puts your process into a wait state for the specified amount of time.

### **PARAMETERS**

#### **time**

A time interval specified in the format hour:minute:second.hundredth, where hour is an integer in the range 0 through 59; minute is an integer in the range 0 through 59; second is an integer in the range 0 through 59; hundredth (of

a second) is an integer in the range 0 through 99. The colons and period are required delimiters. The format is hh:mm:ss.ss.

## **WRITE** logical-name data-item,...

Writes the specified data item as one record to an open file specified by a logical name. All qualifiers must precede all *data-item* expressions.

### **PARAMETERS**

#### **logical-name**

Logical name of the output file. Use the logical name assigned by the OPEN command, or, in interactive mode, specify SYS\$INPUT, SYS\$OUTPUT, SYS\$COMMAND, or SYS\$ERROR to mean your terminal.

#### **data-item**

Symbol name, character string in quotation marks, literal numeric value, or a list of expressions. Multiple data items are concatenated into one record which cannot exceed 255 characters in length.

### **QUALIFIERS**

#### **/ERROR=label**

Transfers control on an I/O error to the location specified by *label* (in a command procedure). /ERROR overrides any ON condition action specified. The symbol \$STATUS retains the error code.

#### **/SYMBOL**

Causes the expression to be interpreted and its expanded value placed in a 2048-byte (instead of a 1024-byte) buffer before the WRITE operation is performed. If you specify multiple expressions, their values are concatenated and placed in the 2048-byte buffer. Each expression specified must be a symbol.

#### **/UPDATE**

Replaces the last record read with the specified *data-item*. The specified data item must be exactly the same size as the record it is replacing. This qualifier can only be used if the file is opened for both READ and WRITE access.



## Appendix DSR

### Digital Standard Runoff

Digital Standard Runoff (DSR) creates formatted files from input files consisting of text, DSR commands, and DSR flags.

#### DSR.1 DCL Commands for Invoking DSR

Process DSR files with the RUNOFF, RUNOFF/CONTENTS, and RUNOFF/INDEX commands. A positional qualifier is local when placed after a file name and global when placed after the command name. A local qualifier applies only to the file it follows; a global qualifier applies to all files unless overridden by a local qualifier.

##### **RUNOFF** dsr-file,...

Creates formatted files from source DSR (RNO) files, unformatted table of contents (RNT) files, and unformatted index (RNX) files. Optionally creates intermediate (BRN) files for input to RUNOFF/CONTENTS and RUNOFF/INDEX commands.

##### PARAMETERS

###### **dsr-file**

Specification of an input DSR file. Wildcard characters are not allowed. The file type defaults to RNO; you must specify the file type for RNT and RNK files. Specify SYS\$INPUT to type the input from your terminal or a command procedure; terminate input from the terminal by pressing CTRL/Z.

##### QUALIFIERS

###### **/BACKSPACE**

Positional qualifier

Bolds, overstrikes, and underlines by backspacing to each character after it is printed, if the printer permits. Otherwise, these operations are implemented by performing a carriage return without a line feed after the entire line is printed.

## **DSR-2     Digital Standard Runoff**

### **DCL Commands for Invoking DSR**

#### **/BOLD[=overstrike] (default)**

##### **/NOBOLD**

Positional qualifier

Specifies the number of times characters are overstruck in a bolding operation. The value must be 0 or a positive integer and defaults to 1. A specification of /BOLD=0 or /NOBOLD disables all bolding, even if the appropriate flags are recognized and enabled.

#### **/CHANGE\_BARS[=character]**

##### **/NOCHANGE\_BARS**

Positional qualifier

Enables change bars (the .BEGIN BAR and .END BAR commands) starting at the beginning of the DSR file (just as if you had included an .ENABLE BAR command within the DSR file). The value specifies the character to be used for the change bar and defaults to a vertical line (|). You can specify the change bar character as a single printable character or a number preceded by a radix indicator (%D, %O, or %X) to represent the ASCII value of a printable or nonprintable character. (For example, you can specify /CHANGE\_BARS=%D7 to ring the bell on terminal output.) A specification of /NOCHANGE\_BARS overrides any .ENABLE BAR command in the DSR file.

#### **/DEBUG[=(option,...)]**

##### **/NODEBUG (default)**

Positional qualifier

Traces certain operations by placing the DSR commands in the output file. The options are as follows:

CONDITIONALS	Ignores conditional commands (.IF, .IFNOT, .ELSE, .ENDIF) in processing the DSR file and places them in the output file.
FILES	Places .REQUIRE commands in the output file (as well as the text of the required file).
INDEX	Places .INDEX and .ENTRY commands in the output file. Index flags are represented by .INDEX commands before the lines on which the flag occurs.
CONTENTS	Places .SEND TOC commands in the output file.

A specification of /DEBUG means /DEBUG=ALL.

#### **/DEVICE[=(option,...)]**

Positional qualifier

Produces an output file suitable for printing on an LN01 or an LN03 printer. The options are as follows.



# Digital Standard Runoff DSR-3

## DCL Commands for Invoking DSR

LN01	Designates a standard LN01 printer with a paper size of 8 1/2 by 11 inches.
LN01E	Designates a European LN01 printer with a paper size of European A4. Incompatible with LN01.
LN03	Designates a standard LN03 printer with a paper size of 8 1/2 by 11 inches.
LANDSCAPE	Prints pages with the long dimension at top using a smaller type size. Allowable page dimensions are 0 to 73 lines per page and 0 to 132 characters per line.
PORTRAIT (default)	Prints pages with the short dimension at top using a larger type size. Allowable page dimensions are 0 to 66 lines per page and 0 to 80 characters per line. Incompatible with LANDSCAPE.
ITALIC (default)	Italicizes characters flagged for underlining. Italicized characters can also be bolded.
UNDERLINE	Underlines characters flagged for underlining. You cannot underline more than 63 consecutive characters (counting a space as a character). Incompatible with ITALIC.

### **/DOWN[=lines]**

#### **/NODOWN (default except for LN01 or LN03)**

Positional qualifier

Shifts the entire page down the specified number of lines. A specification of /DOWN means /DOWN=5, but if you specify /DEVICE=LN01, /DEVICE=LN01E, or /DEVICE=LN03, /DOWN defaults to /DOWN=3.

### **/FORM\_SIZE=lines**

Specifies the maximum number of lines per page including running heads and running feet. Defaults to /FORM\_SIZE=66, which is standard for 11-inch paper. For laser printers, set the number of lines as follows:

Paper size	Lines	Mode
8.05	69	Landscape
8.28	71	Landscape (LN01E default)
8.51	73	Landscape (LN01, LN03 default)
11.00	66	Portrait (LN01, LN03 default)
11.66	70	Portrait (LN01E default)
12.33	74	Portrait

## DSR-4    **Digital Standard Runoff**

### **DCL Commands for Invoking DSR**

Paper size	Lines	Mode
13.00	78	Portrait
14.00	84	Portrait

#### **/INTERMEDIATE[=file-spec]**

##### **/NOINTERMEDIATE (default)**

Positional qualifier

Creates an intermediate file that can be used as input to a table of contents or index operation. The directory and file name default to that of the DSR file. The file type defaults to BRN.

#### **/LOG**

##### **/NOLOG (default)**

Writes a termination message after successful completion of the DSR operation. (The message is always written if the operation fails.) The message states the DSR version number, the number of diagnostic messages (if any), the number of output pages, and the output file specification. If /INTERMEDIATE is specified, the message also includes the number of index records produced and the number of table of contents records produced.

#### **/MESSAGES=(option,...)**

Positional qualifier

Directs error messages to the terminal, to the output file, or to both. You cannot suppress error messages entirely. The options are as follows:

OUTPUT    Output file

USER       Terminal

The default (if you do not specify the qualifier) is /MESSAGES=(OUTPUT,USER).

#### **/OUTPUT[=file-spec] (default)**

##### **/NOOUTPUT**

Positional qualifier

Specifies that an output file is to be produced and optionally names it. The directory and file name default to that of the DSR file. The file type defaults to one of the following:

BLB        For an RNB input file

CCO        For an RNC input file

DOC        For an RND input file

ERR        For an RNE input file

HLP        For an RNH input file

LNI	For an RNO input file with /DEVICE set to LN01, LN01E, or LN03
MAN	For an RNM input file
MEC	For an RNT input file
MEM	For an RNO input file with no /DEVICE specification
MEX	For an RNX input file
OPR	For an RNP input file
PLM	For an RNL input file
STD	For an RNS input file

If you specify /NOOUTPUT, no output file is produced.

**/PAGES="range"**

Positional qualifier

Specifies the pages that will be produced from the output file. Defaults to all pages. Specify the range as follows:

start-page-no:end-page-no,...

You can specify up to five ranges; you can omit the colon and the end page number on the last range to mean the last page. You can omit the quotation marks if you specify only one range. Page numbers must be specified in their default form, not the form specified in a .DISPLAY command. You can specify just the appendix letter or name to produce an entire appendix. You can specify just the word INDEX to produce an entire index.

**/PAUSE**

**/NOPAUSE (default)**

Stops output at the end of each page during processing. You must press the space bar to continue processing. Do not use /PAUSE if you name a spooled device as the output file.

**/REVERSE\_EMPHASIS**

Positional qualifier

Directs DSR to change the order in which flagged text is underlined on an output device. If you use this qualifier, the printer first prints the characters to be underlined, then issues a carriage return without a linefeed, and then prints the underscores to underline the flagged text. If you view your file on the terminal, the flagged characters are overwritten by the underline character.

**/RIGHT[=spaces]**

**/NORIGHT (default except for LN01)**

Positional qualifier

Shifts the text on each page to the right the specified number of spaces. This qualifier does not affect the page width. A specification of /RIGHT means

## **DSR-6    Digital Standard Runoff**

### **DCL Commands for Invoking DSR**

/RIGHT=5. A specification of /RIGHT=0 means /NORIGHT. The defaults (if /RIGHT is not specified) for LN01 files are as follows:

Mode	LN01	LN01E	LN03
Landscape	9	13	9
Portrait	2	2	2

#### **/SEPARATE\_UNDERLINE[=character]**

Positional qualifier

Prints underlines as separate characters on the next line instead of overstriking with underscores on the same line. The value specifies the character to be used for the underline character and defaults to a hyphen (-). You can specify the underline character as a single printable character or a number preceded by a radix indicator (%D, %O, or %X) to represent the ASCII value of a printable or nonprintable character.

#### **/SEQUENCE**

##### **/NOSEQUENCE (default)**

Positional qualifier

Precedes the lines in the output file with the line numbers of the corresponding lines in the DSR file. Sequential numbering is used if line numbers were not explicitly specified in the DSR file.

#### **/SIMULATE**

##### **/NOSIMULATE (default)**

Inserts blank lines instead of a form feed to cause an advance to the top of the next page. Also stops output before the first page is processed; you must press the space bar to continue processing. Do not specify /SIMULATE if you name a spooled device as the output file.

#### **/UNDERLINE\_CHARACTER[=character]**

##### **/NOUNDERLINE\_CHARACTER> RUNOFF**

Positional qualifier

Specifies the character to be used for the underline character. Defaults to an underscore (\_). You can specify the underline character as a single printable character or a number preceded by a radix indicator (%D, %O, or %X) to represent the ASCII value of a printable or nonprintable character. A specification of /NOUNDERLINE\_CHARACTER overrides any .ENABLE UNDERLINING command in the DSR file. Incompatible with /SEPARATE\_UNDERLINE.

**/VARIANT="variant,..."**

Positional qualifier

Identifies conditional structures in the DSR file. You must name conditional structures introduced by .IF to process them. You must name conditional structures introduced by .IFNOT to exclude them. You must not name conditional structures introduced by .ELSE to process them. If you specify only one variant, you do not need the quotation marks.

**RUNOFF/CONTENTS intermediate-file,...**

Creates an unformatted table of contents file from an intermediate file.

**PARAMETERS**

**intermediate-file**

Specification of an input intermediate file. Wildcard characters are not allowed. The file type defaults to BRN. You can concatenate input files into a single output file by connecting the input file specifications with plus signs. (If you separate the input file specifications with commas, separate output files are created.)

**QUALIFIERS**

**/BOLD**

**NOBOLD (default)**

Bolds header titles that are flagged as bold in the text.

**/DEEPEST\_LEVEL=level**

Writes header titles up to the specified level. Level must be an integer in the range 1-6. The default is /DEEPEST\_LEVEL=6.

**/IDENTIFICATION**

**/NOIDENTIFICATION (default)**

Displays the version number of the DSR table of contents facility.

**/INDENT**

**/NOINDENT (default)**

Indents each level of header titles another two spaces. The default indents all levels after the first the same two spaces.

**/LOG**

**/NOLOG (default)**

Writes the name of each input file as it is processed and after it is processed, and the name of each output file created. Error messages are always written.

## **DSR-8     Digital Standard Runoff**

### **DCL Commands for Invoking DSR**

#### **/OUTPUT[=file-spec] (default)**

##### **/NOOUTPUT**

Specifies that an output file is to be produced and optionally names it. The directory and file name default to that of the DSR file. The file type defaults to RNT. If you specify /NOOUTPUT, no output file is produced.

#### **/PAGE\_NUMBERS=(option,...)**

Displays page number references in the table of contents according to the options you specify.

RUNNING

Running page numbers (1, 2, 3, and so on)

NORUNNING (default)

Chapter-oriented page numbers (1-1, 1-2, and so on); incompatible with RUNNING

LEVEL=number

The number of levels for which page numbers are written (specification of LEVEL=0 writes no page numbers); defaults to LEVEL=6

You can specify RUNNING or NORUNNING no matter how the text displays page numbers.

#### **/SECTION\_NUMBERS (default)**

##### **/NOSECTION\_NUMBERS**

Writes section numbers to the table of contents.

#### **/UNDERLINE**

##### **/NOUNDERLINE (default)**

Underlines header titles that are flagged for underlining in the text.

## **RUNOFF/INDEX intermediate-file,...**

Creates a DSR index file from an intermediate file.

### **PARAMETERS**

#### **intermediate-file**

Specification of an input intermediate file. Wildcard characters are not allowed. The file type defaults to BRN. You can concatenate input files into a single output file by connecting the input file specifications with plus signs. (If you separate the input file specifications with commas, separate output files are created.)



## QUALIFIERS

### **/IDENTIFICATION**

#### **/NOIDENTIFICATION (default)**

Displays the version number of the DSR index facility.

### **/LINES\_PER\_PAGE=lines**

Number of lines of index entries on each index page. Defaults to 55 lines. If the original DSR file specifies other than a layout of 0 (.LAYOUT command) and a page length of 58 (.PAGE SIZE command), you should explicitly specify /LINES\_PER\_PAGE as follows: page length minus 3, minus 1 if subtitles are used, minus the number of lines reserved by the layout at the bottom of the page.

### **/LOG**

#### **/NOLOG (default)**

Writes the name of each input file as it is processed and after it is processed, and the name of each output file created. Error messages are always written.

### **/OUTPUT[=file-spec]**

#### **/NOOUTPUT**

Specifies that an output file is to be produced and optionally names it. The directory and file name default to that of the DSR file. The file type defaults to RNX. If you specify /NOOUTPUT, no output file is produced.

### **/PAGE\_NUMBERS=option**

Displays page number references in the index according to the option you specify.

RUNNING

Running page numbers (1, 2, 3, and so on)

NORUNNING (default)

Chapter oriented page numbers (1-1, 1-2, and so on); incompatible with RUNNING

You can specify RUNNING or NORUNNING no matter how the text displays page numbers.

### **/REQUIRE=file-spec**

#### **/NOREQUIRE (default)**

Substitutes a user heading for the standard heading on the first page of the index. The standard heading is the word INDEX centered on the first line, followed by three blank lines. The substitute heading is contained in the file you specify which can contain DSR commands and text. See also /RESERVE.

### **/RESERVE=lines**

#### **/NORESERVE (default)**

Reserves space at the top of the first page of the index for a user heading.

## **DSR.2   DSR Commands**

The DSR commands in the DSR file determine how the text will be formatted in the output file. The format of a DSR command is as follows. The first character of a command must be the control flag, which is a period by default.

`.command-name [parameter,...][;]`

You must observe the following rules:

- **Period in column 1**—The control flag (a period by default) of the first command on a line must be in column 1. No text can precede a command on a line.
- **Multiple commands on one line**—You can place as many DSR commands on one line as space permits except that commands taking text parameters may not be followed by another command.
- **Text on a command line**—You can follow a command with text if you terminate the command with a semicolon. The text must immediately follow the semicolon.
- **Comments**—You can put a comment in your DSR file wherever you can put a command, that is, as the first item on a line or one of a string of commands on a line. Format a comment as follows. The first character of a comment must be the comment flag, which is an exclamation point by default.

`!Any text except a semicolon[;]`

- **Separators**—You must separate the command name from the first parameter and parameters from one another if two consecutive entries are both alphabetic or both numeric. Otherwise, you can choose to separate or not separate the entities. Valid separators between commands and parameters are spaces and tabs. Valid separators between parameters are spaces, tabs, and commas.
- **Abbreviations**—Abbreviations exist for the DSR command names. The abbreviations are listed in parentheses after the command names in the command descriptions that follow.
- **Case**—Command names can be in uppercase or lowercase.
- **Null parameters**—You can enter a null value for a parameter by typing just a comma.

## **.APPENDIX (.AX) [title]**

Starts an appendix by performing the following actions:

1. Issues the commands:  
    .BREAK  
    .LEFT MARGIN 0  
    .SPACING 1  
    .FILL (if .AUTOJUSTIFY is in effect)  
    .JUSTIFY (if .AUTOJUSTIFY is in effect)  
    .PAGING  
    .PAGE
2. Inserts 12 blank lines.
3. Centers on the next line the word APPENDIX followed by a space and the letter or name identifying the appendix.
4. Inserts one blank line.
5. Centers on the next line the specified title. The title is written in uppercase unless case flags indicate otherwise.
6. Inserts three blank lines.

The appendix identifier is regulated by the .NUMBER APPENDIX and .DISPLAY APPENDIX commands.

The title becomes the running head. Any subtitle in effect is blanked.

## **.AUTOJUSTIFY (.AJ)**

Causes the following commands to issue .JUSTIFY and .FILL commands:

.APPENDIX  
.CHAPTER  
.HEADER LEVEL  
.NOTE

Initial default.

## **.AUTOPARAGRAPH (.AP)**

Causes a .PARAGRAPH command to be issued whenever a line begins with a space or a tab. Cancels .AUTOTABLE if it is in effect. The .FILL command must be in effect.

### **.AUTOSUBTITLE (.AST) [header-level]**

Causes .HEADER LEVEL titles to be used for running head subtitles. *Header-level* specifies the highest header level for which subtitles will be written and takes one of the following forms:

- Integer in the range 1 through 6—The exact header level.
- Integer preceded by a plus sign—A value to be added to the last header level specified in an .AUTOSUBTITLE command.
- Integer preceded by a minus sign—A value to be subtracted from the last header level specified in an .AUTOSUBTITLE command.

*Header-level* defaults to 1.

You must issue a .SUBTITLE command for .AUTOSUBTITLE to work. See .SUBTITLE for other effects connected with subtitles.

Initial default: .AUTOSUBTITLE 1

### **.AUTOTABLE (.AT)**

Causes a .PARAGRAPH command to be issued whenever a line does not begin with a space or a tab. Cancels .AUTOPARAGRAPH if it is in effect. The .FILL command must be in effect.

### **.BEGIN BAR (.BB)**

Begins the insertion of change bars at the beginning of lines. The .ENABLE BAR command must be in effect.

### **.BLANK (.B) [lines]**

Issues a .BREAK command and inserts blank lines. Specify *lines* as either:

- Zero or unsigned integer—The number of blank lines to be inserted.
- Integer preceded by a minus sign—The number of lines from the bottom of the page where writing will resume.

*Lines* defaults to 1.

The .BLANK command does not work at the top of a page. (Use the .FIGURE command.) The .BLANK command does not continue to the next page if *lines* is greater than the number of lines left on the page. If the page contains a footnote, the line directly above a footnote is considered the bottom of the page.

### **.BREAK (.BR)**

Ends the current line without filling or justification. A .BREAK command immediately following a .PARAGRAPH, .INDENT, .LEFT MARGIN, .AUTOPARAGRAPH, or .AUTOTABLE command cancels any specified indentation.

### **.CENTER (.C) [line-size]**

Issues a .BREAK command and centers the text that follows. Specify *line-size* as follows:

- Unsigned integer—Twice the value of the position that you want to center the text around.
- Integer preceded by a plus sign—A value to be added to the last centering position used.
- Integer preceded by a minus sign—A value to be subtracted from the last centering position used.

*Line-size* defaults to the value of the left margin plus the value of the right margin, which causes the text to be centered between the two margins.

The .CENTER command must be the last command on a line. Enter the text to be centered on the next line or terminate the command with a semicolon and enter the text to be centered immediately after the semicolon. The text must all be on one line. The line can contain flags but not commands. The text can extend beyond margin and page size settings but cannot extend to the left of position 0.

### **.CHAPTER (.CH) [title]**

Starts a chapter by performing the following actions:

1. Issues the commands:  
    .BREAK  
    .LEFT MARGIN 0  
    .SPACING 1  
    .FILL (if .AUTOJUSTIFY is in effect)  
    .JUSTIFY (if .AUTOJUSTIFY is in effect)  
    .PAGING  
    .PAGE
2. Inserts 12 blank lines.
3. Centers on the next line the word CHAPTER followed by a space and the number of the chapter.
4. Inserts one blank line.

## **DSR-14    Digital Standard Runoff**

### **DSR Commands**

5. Centers on the next line the specified title. The title is written in uppercase unless case flags indicate otherwise.

6. Inserts three blank lines.

The chapter number is regulated by the .NUMBER CHAPTER and .DISPLAY CHAPTER commands.

The title becomes the running head. Any subtitle in effect is blanked.

#### **.COMMENT (!) [text]**

Ignores the text for output processing.

#### **.CONTROL CHARACTERS (.CC)**

Accepts and places in the output file nonprinting characters (characters with ASCII values in the ranges 0 through 31 and 127 through 160).

#### **.DATE (.D)**

Adds the date to running heads. You must issue a .SUBTITLE command for .DATE to work. Either .LAYOUT 1 or .LAYOUT 2 overrides the .DATE command.

#### **.DISABLE BAR (.DBB)**

Disables the .BEGIN BAR and .END BAR commands. If .ENABLE BAR has been in effect, .DISABLE BAR does not shift the lines of text back to their original positions.

Initial default.

#### **.DISABLE BOLDING (.DBO)**

Disables use of the bold flag.

#### **.DISABLE HYPHENATION (.DHY)**

Disables use of the hyphenation flag.



### **.DISABLE INDEXING (.DIX)**

Disables use of the index flag and the commands .INDEX and .ENTRY.

### **.DISABLE OVERSTRIKING (.DOV)**

Disables use of the overstrike flag.

### **.DISABLE TOC (.DTC)**

Disables the collection of information for a table of contents.

### **.DISABLE UNDERLINING (.DUL)**

Disables use of the underline flag.

### **.DISPLAY APPENDIX (.DAX) format-code**

Issues a .BREAK command and defines the appearance of appendix identifiers. Specify *format-code* as one of the following:

D	Decimal numbers
O	Octal numbers
H	Hexadecimal numbers
RU	Uppercase roman numerals
RL	Lowercase roman numerals
RM	Mixed case roman numerals (initial capitals)
LU	Uppercase letters
LL	Lowercase letters
LM	Mixed case letters (initial capitals)

Initial default: .DISPLAY APPENDIX LU

## **.DISPLAY CHAPTER (.DCH) format-code**

Issues a .BREAK command and defines the appearance of chapter identifiers. Specify *format-code* as one of the following:

D	Decimal numbers
O	Octal numbers
H	Hexadecimal numbers
RU	Uppercase roman numerals
RL	Lowercase roman numerals
RM	Mixed case roman numerals (initial capitals)
LU	Uppercase letters
LL	Lowercase letters
LM	Mixed case letters (initial capitals)

Initial default: .DISPLAY CHAPTER D

## **.DISPLAY ELEMENTS (.DLE) ["left",]format-code["right"]**

Issues a .BREAK command and defines the appearance of list-element identifiers. The command must be specified after the .LIST command and before the first .LIST ELEMENT command.

List-element identifiers consist of sequential numbers in the format specified by *format-code*, preceded by the character specified as *left* and followed by the character specified as *right*. *Left*, which must be a single character enclosed in quotation marks or apostrophes, defaults to a space. *Right*, which must be a single character enclosed in quotation marks or apostrophes, defaults to a period. Specify *format-code* as one of the following:

D	Decimal numbers
O	Octal numbers
H	Hexadecimal numbers
RU	Uppercase roman numerals
RL	Lowercase roman numerals
RM	Mixed case roman numerals (initial capitals)
LU	Uppercase letters
LL	Lowercase letters
LM	Mixed case letters (initial capitals)

A .DISPLAY ELEMENTS command remains in effect only until the .END LIST command occurs. The next list uses the default appearance unless another .DISPLAY ELEMENTS command is specified.

Default appearance of lists: .DISPLAY ELEMENTS "" ,D, " ."

### **.DISPLAY LEVELS (.DHL) [format-code],...**

Issues a .BREAK command and defines the appearance of section identifiers in section headers. You can specify or omit up to six *format-code* values, one for each section level. The first *format-code* value corresponds to section level 1, the second *format-code* value to section level 2, and so on. If you omit a *format-code* value, specify the comma unless no more values follow. Specify each *format-code* value as one of the following:

D	Decimal numbers
O	Octal numbers
H	Hexadecimal numbers
RU	Uppercase roman numerals
RL	Lowercase roman numerals
RM	Mixed case roman numerals (initial capitals)
LU	Uppercase letters
LL	Lowercase letters
LM	Mixed case letters (initial capitals)

Initial default: .DISPLAY LEVELS D,D,D,D,D,D

### **.DISPLAY NUMBER (.DNM) format-code**

Issues a .BREAK command and defines the appearance of page numbers starting with the next page number written. Specify *format-code* as one of the following:

D	Decimal numbers
O	Octal numbers
H	Hexadecimal numbers
RU	Uppercase roman numerals
RL	Lowercase roman numerals
RM	Mixed case roman numerals (initial capitals)
LU	Uppercase letters
LL	Lowercase letters
LM	Mixed case letters (initial capitals)

Initial default: .DISPLAY NUMBER D

## DSR-18    Digital Standard Runoff

### DSR Commands

#### **.DISPLAY SUBPAGE (.DSP) format-code**

Issues a .BREAK command and defines the appearance of subpage numbers starting with the next page number written. (Subpage numbers are the numbers appended to the page numbers on subpages, for example, page 1-12A.) Specify *format-code* as one of the following:

D	Decimal numbers
O	Octal numbers
H	Hexadecimal numbers
RU	Uppercase roman numerals
RL	Lowercase roman numerals
RM	Mixed case roman numerals (initial capitals)
LU	Uppercase letters
LL	Lowercase letters
LM	Mixed case letters (initial capitals)

Initial default: .DISPLAY SUBPAGE LU

#### **.ELSE variant**

Starts the *else* portion of a conditional block. *Variant* must be the same value as in the .IF command or .IFNOT command. The *else* portion is processed if the *if* or *ifnot* portion of the conditional block is not processed; otherwise, the *else* portion is not processed. Must be paired with an .IF or .IFNOT command.

#### **.ENABLE BAR (.EBB)**

Enables use of the .BEGIN BAR and .END BAR commands. Causes all further text to be shifted three characters to the right to make room for the bars.

#### **.ENABLE BOLDING (.EBO)**

Enables use of the bold flag.

Initial default.

#### **.ENABLE HYPHENATION (.EHY)**

Enables use of the hyphenation flag.

Initial default.

**.ENABLE INDEXING (.EIX)**

Enables use of the index flag and the commands .INDEX and .ENTRY.

Initial default.

**.ENABLE OVERSTRIKING (.EOV)**

Enables use of the overstrike flag.

Initial default.

**.ENABLE TOC (.ETC)**

Enables the collection of information for a table of contents.

Initial default.

**.ENABLE UNDERLINING (.EUN)**

Enables use of the underline flag.

Initial default.

**.END BAR (.EB)**

Ends the insertion of change bars at the beginning of lines. Must be paired with a .BEGIN BAR command.

**.END FOOTNOTE (.EFN)**

Ends a footnote and restores case, fill, justify, spacing, and margin settings to what they were before the footnote began. Must be paired with a .FOOTNOTE command.

### **.END LIST (.ELS) [lines]**

Ends a list and restores case, fill, justify, spacing, and margin settings to what they were before the list began. Must be paired with a .LIST command. Specify *lines* as one of the following:

- Zero or unsigned integer—The number of blank lines to follow the list.
- Integer preceded by a minus sign—The number of lines from the bottom of the page where writing will resume.

*Lines* defaults to the most recent skip lines setting in a .PARAGRAPH or .SET PARAGRAPH command (parameter 2).

### **.END LITERAL (.EL)**

Ends literal text. Must be paired with a .LITERAL command.

### **.END NOTE (.EN) [lines]**

Ends a note and restores case, fill, justify, spacing, and margin settings to what they were before the note began. Must be paired with a .NOTE command. Specify *lines* as one of the following:

- Zero or unsigned integer—The number of blank lines to follow the note.
- Integer preceded by a minus sign—The number of lines from the bottom of the page where writing will resume.

*Lines* defaults to a value of 1.

### **.END SUBPAGE (.ES)**

Issues the .BREAK command and begins a new page with normal page numbering.

### **.ENDIF variant**

Ends a conditional block. *Variant* must be the same value as in the .IF or .IFNOT command. Must be paired with an .IF or .IFNOT command.

### **.ENTRY (.Y) topic[> subtopic]...**

Creates an index entry without a page reference. The parameters specify the text of the entries. Subtopics are arranged alphabetically under topics.



### **.FIGURE (.FG) [lines]**

Issues the .BREAK command and inserts the number of blank lines specified by *lines*. If the current page does not have sufficient room for all the blank lines, the page is ended and the blank lines are placed on the next page.

*Lines* must be specified as an integer in the range 1 through the maximum number of lines permitted on the page (after header, footer, and forced blank lines are taken into account). *Lines* defaults to a value of 1.

### **.FIGURE DEFERRED (.FGD) [lines]**

Issues the .BREAK command and inserts the number of blank lines specified by *lines*. If the current page does not have sufficient room for all the blank lines, text is added until the page is complete and the blank lines are placed on the next page.

*Lines* must be specified as an integer in the range 1 through the maximum number of lines permitted on the page (after header, footer, and forced blank lines are taken into account). *Lines* defaults to a value of 1.

### **.FILL (.F)**

Causes line endings in the DSR file to be treated as spaces. Lines are created in the output file by accumulating words until the next word would exceed the right margin. The .FILL command also restores: the most recent justification setting set by a .JUSTIFY or .NO JUSTIFY command; any .AUTOPARAGRAPH or .AUTOTABLE setting that was in effect.

Initial default.

### **.FIRST TITLE (.FT)**

Permits a running head to be written on the first page unless a .CHAPTER or .APPENDIX command is issued.

### **.FLAGS ACCEPT (.FL ACCEPT) [character]**

Recognizes the accept flag and optionally specifies a character to represent it. *Character* must be specified as a single character. By default, the accept flag is represented by an underscore (\_).

Initial default.

**DSR-22     Digital Standard Runoff**  
**DSR Commands**

**.FLAGS ALL (.FL)**

Permits recognition of all enabled flags.

Initial default.

**.FLAGS BOLD (.FL BOLD) [character]**

Recognizes the bold flag and optionally specifies a character to represent it. *Character* must be specified as a single character. By default, the bold flag is represented by an asterisk (\*).

**.FLAGS BREAK (.FL BREAK) [character]**

Recognizes the break flag and optionally specifies a character to represent it. *Character* must be specified as a single character. By default, the break flag is represented by a vertical bar (|).

**.FLAGS CAPITALIZE (.FL CAPITALIZE) [character]**

Recognizes the capitalize flag and optionally specifies a character to represent it. *Character* must be specified as a single character. By default, the capitalize flag is represented by a left angle bracket (<).

**.FLAGS COMMENT (.FL COMMENT) [character]**

Recognizes the comment flag and optionally specifies a character to represent it. *Character* must be specified as a single character. By default, the comment flag is represented by an exclamation point (!).

Initial default.

**.FLAGS CONTROL (.FL CONTROL) [character]**

Recognizes the control flag and optionally specifies a character to represent it. *Character* must be specified as a single character. By default, the control flag is represented by a period (.).

Initial default.

**.FLAGS HYPHENATE (.FL HYPHENATE) [character]**

Recognizes the hyphenate flag and optionally specifies a character to represent it. *Character* must be specified as a single character. By default, the hyphenate flag is represented by an equal sign (=).

**.FLAGS INDEX (.FL INDEX) [character]**

Recognizes the index flag and optionally specifies a character to represent it. *Character* must be specified as a single character. By default, the index flag is represented by a right angle bracket (>).

**.FLAGS LOWERCASE (.FL LOWERCASE) [character]**

Recognizes the lowercase flag and optionally specifies a character to represent it. *Character* must be specified as a single character. By default, the lowercase flag is represented by a backslash (\).

Initial default.

**.FLAGS OVERSTRIKE (.FL OVERSTRIKE) [character]**

Recognizes the overstrike flag and optionally specifies a character to represent it. *Character* must be specified as a single character. By default, the overstrike flag is represented by a percent sign (%).

**.FLAGS PERIOD (.FL PERIOD) [character]**

Recognizes the period flag and optionally specifies a character to represent it. *Character* must be specified as a single character. By default, the period flag is represented by a plus sign (+).

**.FLAGS SPACE (.FL SPACE) [character]**

Recognizes the space flag and optionally specifies a character to represent it. *Character* must be specified as a single character. By default, the space flag is represented by a number sign (#).

Initial default.

**.FLAGS SUBINDEX (.FL SUBINDEX) [character]**

Recognizes the subindex flag and optionally specifies a character to represent it. *Character* must be specified as a single character. By default, the subindex flag is represented by a right angle bracket ( > ).

Initial default.

**.FLAGS SUBSTITUTE (.FL SUBSTITUTE) [character]**

Recognizes the substitute flag and optionally specifies a character to represent it. *Character* must be specified as a single character. By default, the substitute flag is represented by a dollar sign ( \$ ).

**.FLAGS UNDERLINE (.FL UNDERLINE) [character]**

Recognizes the underline flag and optionally specifies a character to represent it. *Character* must be specified as a single character. By default, the underline flag is represented by an ampersand ( & ).

Initial default.

**.FLAGS UPPERCASE (.FL UPPERCASE) [character]**

Recognizes the uppercase flag and optionally specifies a character to represent it. *Character* must be specified as a single character. By default, the uppercase flag is represented by a circumflex ( ^ ).

Initial default.

**.FOOTNOTE (.FN)**

Places the text that follows up to the .END FOOTNOTE command at the bottom of the page if it fits, or at the bottom of the next page if it does not fit. Must be paired with an .END FOOTNOTE command.

The .FOOTNOTE command does not provide any special formatting. Format the footnote by including DSR commands within the footnote.

If the .NO PAGING command is in effect, all footnotes appear at the end of the document.

## **.HEADER LEVEL (.HL) [level] [title]**

Starts a section by performing the following actions:

1. Issues the commands:
  - .BREAK
  - .TEST PAGE lines (as specified in .STYLE HEADER)
  - .SPACING 1
  - .FILL (if .AUTOJUSTIFY is in effect)
  - .JUSTIFY (if .AUTOJUSTIFY is in effect)
2. Writes the section number and header, formatted according to the .STYLE HEADER values in effect.
3. Inserts punctuation and/or spacing according to the .STYLE HEADER values in effect.

*Level* must be specified as one of the following:

- Unsigned integer in the range 1 to 6—The exact level of the header.
- Integer preceded by a plus sign—A value to be added to the last level specified in a .HEADER LEVEL or .SET LEVEL command (or 1 if a level has not yet been set).
- Integer preceded by a minus sign—A value to be subtracted from the last level specified in a .HEADER LEVEL or .SET LEVEL command (or 1 if a level has not yet been set).

*Level* defaults to the value of the last level specified in a .HEADER LEVEL or .SET LEVEL command (or 1 if the level has not yet been set).

The title must follow on the same line as the command with no intervening semicolon. If the title exceeds the size of the output line, the first line is filled (and justified if justification is in effect) and the title continues on the next line.

The title becomes the running head. Any subtitle in effect is blanked.

## **.HEADERS LOWER (.HD LOWER)**

Writes in lowercase the word “page” that precedes the page number in some layouts.  
Writes in lowercase the word “index” that is part of the page number in indexes.

### **.HEADERS MIXED (.HD MIXED)**

Writes in lowercase with an initial capital the word "Page" that precedes the page number in some layouts. Writes in lowercase with an initial capital the word "Index" that is part of the page number in indexes.

Initial default.

### **.HEADERS [ON] (.HD)**

Writes running heads on each page except the first (unless you also specify .FIRST TITLE) according to the current .LAYOUT values.

Initial default.

### **.HEADERS UPPER (.HD UPPER)**

Writes in uppercase the word "PAGE" that precedes the page number in some layouts. Writes in uppercase the word "INDEX" that is part of the page number in indexes.

### **.IF variant**

Starts a conditional block and introduces the *if* portion of the block. The *if* portion of the block is processed if *variant* is specified as a value of the /VARIANT qualifier in the invoking DSR command. The .IF command must be paired with an .ENDIF command. The conditional block can contain an .ELSE command.

You can nest one conditional block within another. The nested conditional block must be entirely contained within the *if* or the *else* portion of the nesting block.

### **.IFNOT variant**

Starts a conditional block and introduces the *ifnot* portion of the block. The *ifnot* portion of the block is processed if *variant* is not specified as a value to the /VARIANT qualifier in the invoking DSR command. The .IFNOT command must be paired with an .ENDIF command and can be paired with an .ELSE command. The .ELSE command must precede the .ENDIF command.

You can nest one conditional block within another. The nested conditional block must be entirely contained within the *if* or the *else* portion of the nesting block.



### **.INDENT (.I) spaces**

Issues a .BREAK command and indents a line of text. If you specify a .BREAK command after .INDENT, the indent operation is canceled. Specify *spaces* as one of the following:

- Zero or unsigned integer—The number of spaces to indent to the right of the left margin.
- Integer preceded by a minus sign—The number of spaces to indent to the left of the left margin. You cannot indent past position 0.

The parameter defaults to the value of the *spaces* parameter of the most recent .PARAGRAPH or .SET PARAGRAPH command.

You can enter the line of text after the command (with an intervening semicolon) or on the next line. The line cannot contain commands.

### **.INDEX (.X) topic[> subtopic...]**

Creates an index entry with a page reference. The parameters specify the text of the entries. Subtopics are arranged alphabetically under topics with their own page references.

### **.JUSTIFY (.J)**

Issues a .BREAK command and causes the text that follows to be justified—extra spaces are inserted between words so that the last character on each line reaches the right margin.

Initial default.

### **.KEEP (.K)**

Causes blank lines in the input file to be inserted in the output file. The .NO FILL command must be in effect.

## **.LAYOUT (.LO) code [,lines]**

Issues a .BREAK command and specifies the format for page header and footer information. Specify *code* as one of the following:

Code	Description
0	Running head appears in the upper left of the page. Page number and date appear in the upper right.
1	Running head appears centered at the top of the page. Page number appears centered at the bottom.
2	Running head appears at the top right of an odd-numbered page and the top left of an even-numbered page. Page number appears centered at the bottom.
3	Running head appears in the upper left of the page. Date appears in the upper right. Page number appears centered between hyphens at the bottom of the page. Page numbers are consecutive through the entire document.

If code equals 1, 2, or 3, specify *lines* as an unsigned integer equal to the number of lines below the last line of text that the page number will appear. If code equals 0, do not specify *lines*.

Initial default: .LAYOUT 0

## **.LEFT MARGIN (.LM) position**

Issues a .BREAK command and sets the position of the left margin. Specify *position* as one of the following:

- Unsigned integer—The exact position of the left margin.
- Integer preceded by a plus sign—A value to be added to the current position of the left margin.
- Integer preceded by a minus sign—A value to be subtracted from the current position of the left margin.

*Position* defaults to a value of 0.

The left margin must be greater than 0 and less than the right margin.

Initial default: .LEFT MARGIN 0

### **.LIST (.LS) [lines] [,"character"]**

Starts a list by performing the following actions:

1. Issues a .BREAK command.
2. Issues the command .LEFT MARGIN 9 if the left margin is currently 0; otherwise, issues the command .LEFT MARGIN +4.
3. Issues a .TEST PAGE command specifying for the *lines* parameter a value of 2 plus the value of *test-lines* in the most recent .PARAGRAPH or .SET PARAGRAPH command.

Specify *lines* as one of the following:

- Zero or unsigned integer—The number of blank lines to be inserted before each element in the list.
- Integer preceded by a minus sign—The number of lines from the bottom of the page where writing will resume.

Specify *character* as a character enclosed in quotation marks or apostrophes. Each list element is preceded by this character and two spaces. If you omit *character*, the list elements are preceded by a sequence of numbers starting with number 1.

The .LIST command must be paired with an .END LIST command. A list can contain .LIST ELEMENT commands. You can nest one list within another to a maximum level of 14; a nested list must be entirely contained within one element of the nesting list.

### **.LIST ELEMENT (.LE)**

Issues a .BREAK command and writes the text that follows as a list element. The text is terminated by either another .LIST ELEMENT command or an .END LIST command.

You can specify .LIST ELEMENT without having issued a .LIST command. List elements not enclosed in lists begin at the left margin and are numbered sequentially throughout the entire document. Do not issue an .END LIST command.

## **DSR-30    Digital Standard Runoff**

### **DSR Commands**

#### **.LITERAL (.LT)**

Issues a .BREAK command, issues the command .RIGHT MARGIN 150, and writes text as it appears in the input file until an .END LITERAL command occurs. Must be paired with an .END LITERAL command. When .LITERAL is in effect, flags are treated as text even if they are recognized and enabled. Commands and flags in effect before the .LITERAL command are disabled (until the .END LITERAL command occurs) except for .LEFT MARGIN, .TAB STOPS, the underline flag, and the bold flag.

#### **.NO AUTOJUSTIFY (.NAJ)**

Disables the effects of the .AUTOJUSTIFY command.

#### **.NO AUTOPARAGRAPH (.NAP)**

Disables the effects of the .AUTOPARAGRAPH command.

Initial default.

#### **.NO AUTOSUBTITLE (.NAST)**

Disables the effects of the .AUTOSUBTITLE command.

#### **.NO AUTOTABLE (.NAT)**

Disables the effects of the .AUTOTABLE command.

Initial default.

#### **.NO CONTROL CHARACTERS (.NCC)**

Disables the effects of the .CONTROL CHARACTERS command.

Initial default.

#### **.NO DATE (.ND)**

Disables the effects of the .DATE command.

Initial default.

**.NO FILL (.NF)**

Issues a .BREAK command and disables the effects of the .AUTOPARAGRAPH, .AUTOTABLE, .FILL, and .JUSTIFY commands.

**.NO FLAGS ACCEPT (.NFL ACCEPT)**

Disables recognition of the accept flag.

**.NO FLAGS [ALL] (.NFL [ALL])**

Does not permit recognition of flags except the comment and control flags.  
Initial default.

**.NO FLAGS BOLD (.NFL BOLD)**

Disables recognition of the bold flag.  
Initial default.

**.NO FLAGS BREAK (.NFL BREAK)**

Disables recognition of the break flag.  
Initial default.

**.NO FLAGS CAPITALIZE (.NFL CAPITALIZE)**

Disables recognition of the capitalize flag.  
Initial default.

**.NO FLAGS COMMENT (.NFL COMMENT)**

Disables recognition of the comment flag.

**.NO FLAGS CONTROL (.NFL CONTROL)**

Disables recognition of the control flag.

**DSR-32    Digital Standard Runoff**  
**DSR Commands**

**.NO FLAGS HYPHENATE (.NFL HYPHENATE)**

Disables recognition of the hyphenate flag.

Initial default.

**.NO FLAGS INDEX (.NFL INDEX)**

Disables recognition of the index flag.

Initial default.

**.NO FLAGS LOWERCASE (.NFL LOWERCASE)**

Disables recognition of the lowercase flag.

**.NO FLAGS OVERSTRIKE (.NFL OVERSTRIKE)**

Disables recognition of the overstrike flag.

Initial default.

**.NO FLAGS PERIOD (.NFL PERIOD)**

Disables recognition of the period flag.

Initial default.

**.NO FLAGS SPACE (.NFL SPACE)**

Disables recognition of the space flag.

**.NO FLAGS SUBINDEX (.NFL SUBINDEX)**

Disables recognition of the subindex flag.

**.NO FLAGS SUBSTITUTE (.NFL SUBSTITUTE)**

Disables recognition of the substitute flag.

Initial default.



**.NO FLAGS UNDERLINE (.NFL UNDERLINE)**

Disables recognition of the underline flag.

**.NO FLAGS UPPERCASE (.NFL UPPERCASE)**

Disables recognition of the uppercase flag.

**.NO JUSTIFY (.NJ)**

Issues a .BREAK command and disables the effects of the .JUSTIFY command.

**.NO KEEP (.NK)**

Disables the effects of the .KEEP command.

Initial default.

**.NO NUMBER (.NNM)**

Suspends the writing of page numbers, unless .LAYOUT 3 is in effect.

**.NO PAGING (.NPA)**

Causes the document to be written without page breaks and without reserving room for headers and footers.

**.NO PERIOD (.NPR)**

Disables the effects of the .PERIOD command.

**.NO SPACE (.NSP)**

Causes a space not to be inserted in place of a carriage return between two lines of text. The .NO SPACE command must be placed between two lines of text and affects only those lines. The .FILL command must be in effect.

**.NO SUBTITLE (.NST)**

Causes subtitles not to be written.

Initial default.

**.NOTE (.NT) [title]**

Performs the following actions:

1. Issues a .BREAK command.
2. Issues a .TEST PAGE command specifying for the *lines* parameter the value of *test-lines* in the most recent .PARAGRAPH or .SET PARAGRAPH command.
3. Issues the command .SKIP 1, writes the specified title, and issues the command .SKIP 1. The title defaults to the word NOTE in uppercase.
4. If the left margin is set to position 0, issues the commands .LEFT MARGIN +8 and .RIGHT MARGIN -8. Otherwise, issues the commands LEFT MARGIN +4 and .RIGHT MARGIN -4.
5. Issues the command .FILL. Issues the command .JUSTIFY if autojustification is in effect. Writes the text that follows the .NOTE command until an .END NOTE command occurs.

A .NOTE command must be paired with an .END NOTE command.

**.NUMBER APPENDIX (.NMAX) [identifier]**

Specifies the identifier of the next appendix as follows:

- Character—A single character.
- Character string—Up to five characters.
- Unsigned integer—The sequence number of the letter in the alphabet: 1 means A, 2 means B, 26 means Z, 27 means AA, and so on.
- Integer preceded by a plus sign—A value to be added to the sequence number of the current identifier.
- Integer preceded by a minus sign—A value to be subtracted from the sequence number of the current identifier.

*Identifier* defaults to a value of A.

If you do not explicitly provide an identification for the next appendix with the .NUMBER APPENDIX command, the identifier assumes the value of the identifier of the current appendix incremented by one.

Initial default: .NUMBER APPENDIX A

### **.NUMBER CHAPTER (.NMCH) number**

Specifies the identifier of the next chapter as follows:

- Unsigned integer—The number of the chapter.
- Integer preceded by a plus sign—A value to be added to the number of the current chapter.
- Integer preceded by a minus sign—A value to be subtracted from the number of the current chapter.

*Number* defaults to a value of 1.

If you do not explicitly provide an identification for the next chapter with the .NUMBER CHAPTER command, the chapter assumes the value of the current chapter number incremented by one.

Initial default: .NUMBER CHAPTER 1

### **.NUMBER LEVEL (.NMLV) [number],...**

Specifies a base section number. You can specify up to six numbers: each number corresponds to a level. Omit a level by including just the comma, except that trailing commas can be omitted. A number defaults to the current number in effect for the level. Specify each number as follows:

- Unsigned integer—The number of the section for the indicated level.
- Integer preceded by a plus sign—A value to be added to the current number for the indicated level.
- Integer preceded by a minus sign—A value to be subtracted from the current number for the indicated level.

If you do not explicitly provide a number for the next section header with the .NUMBER LEVEL command, the section header number assumes the value of the current section header number incremented by one.

### **.NUMBER LIST (.NMLS) number**

Specifies the number of the next element in a list.

If you do not explicitly provide a number for the next list element with the .NUMBER LIST command, the list element number assumes the value of the current list element number incremented by one.

### **.NUMBER [PAGE] (.NMPG) [number]**

Resumes page numbering if .NO NUMBER is in effect. Sets the number of the next page to the value of *number*, which must be one of the following:

- Unsigned integer—The number of the next page.
- Integer preceded by a plus sign—A value to be added to the current page number.
- Integer preceded by a minus sign—A value to be subtracted from the current page number.

*Number* defaults to the current page number.

Do not use the .NUMBER PAGE command if the .LAYOUT 3 command is in effect.

### **.NUMBER RUNNING (.NMR) number**

Sets the number of the next page to the value of *number*, which must be one of the following:

- Unsigned integer—The number of the next page.
- Integer preceded by a plus sign—A value to be added to the current page number.
- Integer preceded by a minus sign—A value to be subtracted from the current page number.

*Number* defaults to the current page number.

Use only if the .LAYOUT 3 command is in effect.

### **.NUMBER SUBPAGE (.NMSPG) [identifier]**

Specifies the identifier of the next subpage as follows:

- One or more characters—A letter such as A, B, Z, or AA.
- Unsigned integer—The sequence number of the letter in the alphabet: 1 means A, 2 means B, 26 means Z, 27 means AA, and so on.
- Integer preceded by a plus sign—A value to be added to the sequence number of the current identifier.
- Integer preceded by a minus sign—A value to be subtracted from the sequence number of the current identifier.

*Identifier* defaults to a value of A.

The .NUMBER SUBPAGE command issues a .SUBPAGE command if .SUBPAGE is not already in effect.

### **.PAGE (.PG)**

Issues a .BREAK command and starts a new page. The current page must contain at least one line of text. The .PAGE command works even if .NO PAGING is in effect.

### **.PAGE SIZE (.PS) [max-lines], [running-width]**

Issues a .BREAK command, sets a maximum for the number of lines of text on a page, and sets the page width for writing running heads (does not affect normal text). Specify *max-lines* as follows:

- Unsigned integer—The maximum number of lines allowed.
- Integer preceded by a plus sign—A value to be added to the current maximum.
- Integer preceded by a minus sign—A value to be subtracted from the current maximum.

*Max-lines* defaults to the current maximum. The maximum cannot be less than 13.

Specify *running-width* as follows:

- Unsigned integer—The width of the page for writing running heads.
- Integer preceded by a plus sign—A value to be added to the current width.
- Integer preceded by a minus sign—A value to be subtracted from the current width.

*Running-width* defaults to the current width. The width cannot exceed 150.

## DSR-38    Digital Standard Runoff

### DSR Commands

If the .NO PAGING command is in effect, the .PAGE SIZE command issues a .PAGING command.

Initial default: .PAGE SIZE 58,70

### **.PAGING (.PA)**

Issues a .BREAK command and causes the document to be split into pages.

Initial default.

### **.PARAGRAPH (.P) [spaces],[skip-lines],[test-lines]**

Issues a .BREAK command followed by .TEST PAGE, .SKIP, and .INDENT commands. Specify *spaces* as one of the following:

- Zero or unsigned integer—The number of spaces to indent the first line of the paragraph to the right of the left margin.
- Integer preceded by a minus sign—The number of spaces to indent to the left of the left margin. You cannot indent past position 0.

Specify *skip-lines* as one of the following:

- Zero or unsigned integer—The number of blank lines to be inserted before the paragraph.
- Integer preceded by a minus sign—The number of lines from the bottom of the page where writing will resume.

Specify *test-lines* as a factor in determining the number of lines that must be left on the page for the paragraph to start on the page. Otherwise, the paragraph starts on the next page. The precise algorithm is as follows, where *spacing* is the current value of the .SPACING command (initial default is 1):

$$(\text{skip-lines} + \text{test-lines} + 1) * \text{spacing}$$

All three parameters default to the values specified in the last .PARAGRAPH or .SET PARAGRAPH command.

Initial default: .PARAGRAPH 5,1,2



### **.PERIOD (.PR)**

Adds an extra space after a period (.), colon (:), question mark (?), or exclamation point (!). The .FILL command must be in effect, the character must be followed by a space or carriage return, and the character must not be preceded by the accept flag.

Initial default.

### **.REPEAT (.RPT) times, "characters"**

Repeats the specified characters the specified number of times. The characters must be enclosed in quotation marks or apostrophes. You cannot specify more than 150 characters.

If the .FILL command is in effect, the characters are repeated horizontally. If the .NOFILL command is in effect, the characters are repeated vertically starting at the left margin.

### **.REQUIRE (.REQ) "file-spec"**

Processes the contents of another file as if those contents existed in place of the .REQUIRE command. The file specification must be enclosed in quotation marks or apostrophes. The file type defaults to RNO. A .REQUIRE command must be the last command on a line.

### **.RESTORE (.RE)**

Restores the formatting context saved by the last .SAVE command. Must be paired with a .SAVE command.

### **.RIGHT (.R) [spaces]**

Issues a .BREAK command and positions a line of text relative to the right margin. If you specify a .BREAK command after the .RIGHT, the operation is canceled. Specify *spaces* as one of the following:

- Zero or an unsigned integer—The number of spaces to indent to the left of the right margin. You cannot indent past position 0.
- Integer preceded by a minus sign—The number of spaces to extend to the right of the right margin.

*Spaces* defaults to 0.

## DSR-40    Digital Standard Runoff

### DSR Commands

You can enter the line of text after the command (with an intervening semicolon) or on the next line. The line cannot contain commands.

#### **.RIGHT MARGIN (.RM) [position]**

Issues a .BREAK command and sets the position of the right margin. Specify *position* as one of the following:

- Unsigned integer—The exact position of the right margin.
- Integer preceded by a plus sign—A value to be added to the current position of the right margin.
- Integer preceded by a minus sign—A value to be subtracted from the current position of the right margin.

*Position* defaults to 70.

Initial default: .RIGHT MARGIN 70

#### **.SAVE (.SA)**

Saves the current formatting context which includes the following settings: date status, fill, flags, headers, justification, keep status, margins, numbering, page size, paging, paragraph parameters, spacing, subtitles, and tab stops. The .SAVE command must be paired with a .RESTORE command. You can nest one saved-formatting context within another to a depth of 20.

#### **.SEND TOC (.STC) toc-line**

Writes a line to the table of contents. The line can contain text, DSR commands, and DSR flags.

#### **.SET DATE (.SDT) [[day],[month],[year]]**

Issues a .BREAK command and sets the date. Specify *day*, *month*, or *year* as follows:

- Unsigned integer—The day of the month, month, or year. You can specify the entire year or the last two digits.
- Integer preceded by a plus sign—A value to be added to the current value of *day*, *month*, or *year*.
- Integer preceded by a minus sign—A value to be subtracted from the current value of *day*, *month*, or *year*.

You can omit a parameter by including only the comma. Omitted parameters default to the current values.

If you omit the parameters entirely, the date is reset to the current date (the date on which the DSR processing occurs).

Initial default: .SET DATE current-date

### **.SET LEVEL (.SL) [level]**

Sets the level used in .HEADER LEVEL commands. Specify level as one of the following:

- Unsigned integer in the range 1-6—The exact level of the header.
- Integer preceded by a plus sign—A value to be added to the current level setting.
- Integer preceded by a minus sign—A value to be subtracted from the current level setting.

*Level* defaults to the last level specified in a .HEADER LEVEL or .SET LEVEL command (or 1 if a level has not yet been set).

### **.SET PARAGRAPH (.SPR) [spaces],[skip-lines],[test-lines]**

Sets the parameter values used in .PARAGRAPH commands. Specify *spaces* as one of the following:

- Zero or unsigned integer—The number of spaces to indent the first line to the right of the left margin.
- Integer preceded by a minus sign—The number of spaces to indent to the left of the left margin. You cannot indent past position 0.

Specify *skip-lines* as one of the following:

- Zero or unsigned integer—The number of blank lines to be inserted before the paragraph.
- Integer preceded by a minus sign—The number of lines from the bottom of the page where writing will resume.

Specify *test-lines* as a factor in determining the number of lines that must be left on the page for the paragraph to start on the page. Otherwise, the paragraph starts on the next page. The precise algorithm is as follows, where *spacing* is the current value of the .SPACING command (initial default is 1).

$(\text{skip-lines} + \text{test-lines} + 1) * \text{spacing}$

## DSR-42    Digital Standard Runoff

### DSR Commands

The parameters default to the values specified in the last .PARAGRAPH or .SET PARAGRAPH command.

#### **.SET TIME (.STM) [[hour],[minute],[second]]**

Issues a .BREAK command and sets the time. Specify *hour*, *minute*, or *second* as follows:

- Unsigned integer—The hour of the day, minute of the hour, or second of the minute.
- Integer preceded by a plus sign—A value to be added to the current hour, minute, or second.
- Integer preceded by a minus sign—A value to be subtracted from the current hour, minute, or second.

You can omit parameters by including only the comma. Omitted parameters default to the current values.

If you omit the parameters entirely, the time is reset to the current time (the time at which the DSR processing occurs).

Initial default: .SET TIME current-time

#### **.SKIP (.S) [lines]**

Issues a .BREAK command and inserts blank lines. *Lines* must be one of the following. The value is multiplied by the current value of the .SPACING command (initial default is 1).

- Zero or unsigned integer—The number of blank lines to be inserted.
- Integer preceded by a minus sign—The number of lines from the bottom of the page where writing will resume.

*Lines* defaults to a value of 1.

The .SKIP command does not work at the top of a page unless *lines* is negative. (Use the .FIGURE command.) The .SKIP command does not continue to the next page if *lines* is greater than the number of lines left on the page. If the page contains a footnote, the line directly above the footnote is considered the bottom of the page.

### **.SPACING (.SP) lines**

Sets the spacing between lines of text, where 1 means single spacing (no blank lines between lines of text), 2 means double spacing, and so on. The following commands multiply a *lines* specification by the current .SPACING value to determine the number of physical lines: .AUTOPARAGRAPH, .AUTOTABLE, .PARAGRAPH, .SET PARAGRAPH, and .SKIP.

Specify *lines* as an integer in the range 1 through 5.

Initial default: .SPACING 1

### **.STYLE HEADERS (.STHL) [code],...**

Issues a .BREAK command and sets the formatting for level headers. You can specify up to nine codes as follows. The codes must be written in the specified position. Specify a default value by including just the comma.

1. Run-in format—An integer in the range 0 through 7 specifying the lowest numbered level of header to have a run-in format. (A run-in format starts the text on the same line as the header.) Defaults to 3.
2. Uppercase title—An integer in the range 0 through 7 specifying the highest numbered level of header to have its title written in all uppercase letters. Overrides any conflicting code 3 setting. Defaults to 1.
3. Initial capitals in title—An integer in the range 0 through 7 specifying the highest numbered level of header to have its title written in uppercase and lowercase (that is, initial capitals). Defaults to 6.
4. Section number—An integer in the range 0 through 7 specifying the lowest numbered level of header not to have a section number written to the left of the title. Defaults to 7.
5. Centered title—An integer in the range 0 through 7 specifying the lowest numbered level of header to have its header centered. (However, run-in headers are not centered.) Defaults to 7.
6. Preceding blank lines—An unsigned integer specifying the number of blank lines preceding a header. Defaults to 2.
7. Following blank lines—An unsigned integer specifying the number of blank lines following a header. Defaults to 1.
8. Test page lines—An unsigned integer specifying the number of lines that must be left on a page for the level header to be written to that page. Otherwise, the header is written at the top of the next page. Defaults to 7 plus the most recent value specified for the test-lines parameter of a .PARAGRAPH or .SET PARAGRAPH command.



## DSR-44    Digital Standard Runoff

### DSR Commands

9. Spaces after section number—An integer in the range 1 through 75 specifying the number of spaces between the header number and title. Defaults to 2.

Initial default: .STYLE HEADERS 3,1,6,7,7,2,1,9,2

### **.SUBPAGE (.SPG)**

Issues a .BREAK command and starts a new page. Initiates subpage numbering: each page has the number of the most recent page immediately followed by a subpage appendix. For example, page 1-12 might be followed by subpages 1-12A and 1-12B. Must be paired with an .END SUBPAGE command.

### **.SUBTITLE (.ST) [subtitle]**

Issues a .BREAK command and sets *subtitle* as the running head to be used. *Subtitle* must be specified as a character string. A .SUBTITLE command must be the last command on a line.

### **.TAB STOPS (.TS) [position],...**

Sets the tab stops. Specify *position* as one of the following:

- Unsigned integer—The exact position of the tab stop.
- Integer preceded by a plus sign—A value to be added to the current position of the tab stop.
- Integer preceded by a minus sign—A value to be subtracted from the current position of the tab stop.

Tab stops are set left to right as specified. Each tab stop must be at least two greater than the preceding tab stop. You can retain the value of the previous tab stop by specifying just a comma; you must specify the commas to retain trailing tab stops.

Initial default: .TAB STOPS 8,16,24,...

### **.TEST PAGE (.TP) lines**

Issues a .BREAK command and checks the current page for the specified number of remaining lines. If the page does not contain enough room for the specified number of lines, a new page is started. *Lines* must be specified as an unsigned integer.



### **.TITLE (.T) [title]**

Issues a .BREAK command and sets *title* as the running head title to be used. *Title* must be specified as a character string. A .TITLE command must be the last command on a line.

### **.VARIABLE (.VR) name [true, false]**

Identifies text and commands contained in conditional blocks if the /DEBUG qualifier is specified with the RUNOFF command.

*Name* must be the name of a variable in an .IF or .IFNOT command. The .IF or .IFNOT command must follow the .VARIABLE command.

*True* must be a single character. This character (followed by a space) will appear as the first character of each line which is true for the specified variable.

*False* must be a single character. This character (followed by a space) will appear as the first character of each line which is false for the specified variable.

### **.XLOWER (.XL)**

Uppercases and lowercases index entries exactly as they appear in the DSR file.

### **.XUPPER (.XU)**

Uppercases the first character of an index entry and lowercases the remaining characters, except where explicitly overridden by the capitalize or uppercase flag.

Initial default.

## **DSR.3 DSR Flags**

The following conditions must be met for a flag to be used:

- Master recognition—The .FLAGS ALL command must be in effect, except for the comment and control flags.
- Individual recognition—The specific .FLAGS command must be in effect.
- Individual enabling—The specific .ENABLE command must be in effect.

If a character is not recognized as a flag, the character is treated as normal text. If a character is not enabled as a flag, the character does not cause the flag action to occur.

## DSR-46 Digital Standard Runoff

### DSR Flags

The flags are as follows. The second column lists the initial default character representing each flag.

Flag Name	D	Description
ACCEPT	—	Treats the next character as text even if the character is a flag that is in effect, a period, or a space.
BOLD	*	Writes the next character in bold (by overstriking).
BREAK		Permits a word to be broken between lines at the point where the flag appears. No hyphen is automatically inserted.
CAPITALIZE	<	Capitalizes all characters that follow until one of the following occurs: an expandable space, a BREAK flag, a HYPHENATE flag, a CAPITALIZE flag, a pair of UPPERCASE flags, a pair of LOWERCASE flags, the end of the line.
COMMENT	!	Treats the characters that follow as a comment until a semicolon or the end of the line occurs. A comment can only occur where a DSR command is legal; otherwise, the COMMENT flag is treated as text.
CONTROL	.	Treats the characters that follow as a DSR command until a semicolon or the end of the line occurs. The CONTROL flag is treated as text if it occurs where a DSR command is not legal.
HYPHENATE	=	Permits a word to be broken between lines at the point where the flag appears. A hyphen is automatically inserted.
INDEX	>	Treats the characters that follow as an index entry.
LOWERCASE	\	Lowercases the next character.
OVERSTRIKE	%	Overstrikes the preceding character with the next character.
PERIOD	+	Inserts an expandable space when the flag is placed after the last character in a word.
SPACE	#	Produces one unexpandable space. The word preceding the flag, the flag, and the next word are all treated as one word in processing.
SUBINDEX	>	In an .INDEX command, treats the characters that follow as a subentry. In an .ENTRY command, treats the characters that follow as a cross-reference.
SUBSTITUTE	\$	Must be paired with itself. See the next table.
UNDERLINE	&	Underlines the next character.
UPPERCASE	^	Uppercases the next character.

The following flag pairs can be used. The second column lists the default characters representing each flag pair.

Flag Pair	D	Description
CONTROL COMMENT	.!	Makes an entire line a comment.
CONTROL CONTROL	..	Inserts an actual period in the text.
LOWERCASE BOLD	\*	Ends the bolding of characters.
LOWERCASE LOWERCASE	\\	Writes the text that follows in lowercase until another case flag occurs.
LOWERCASE UNDERLINE	\&	Ends the underlining of characters.
SUBSTITUTE SUBSTITUTE	\$\$	Inserts the date, time, or a part of the date or time depending on a keyword that follows the SUBSTITUTE SUBSTITUTE flag pair. The keyword must immediately follow the flag pair and can be uppercase or lowercase. The valid keywords are as follows: DATE, TIME, YEAR, MONTH, DAY, HOURS, MINUTES, SECONDS.
UNDERLINE SPACE	&#	Inserts an actual underscore into the text.
UPPERCASE BOLD	^*	Bolds the characters that follow until a LOWERCASE BOLD flag pair occurs.
UPPERCASE CAPITALIZE	^ <	Capitalizes the characters that follow until another case flag occurs.
UPPERCASE UNDERLINE	^&	Underlines the characters that follow until a LOWERCASE UNDERLINE flag pair occurs.
UPPERCASE UPPERCASE	^^	Writes the text that follows as it appears in the DSR file until another case flag occurs.

## DSR.4 Index Formatting

Indexes are formatted according to the following rules:

- **Punctuation**—A comma separates an index entry or subentry from its page reference. Commas separate multiple page references for the same entry. No comma follows an entry that does not have a page reference.
- **Position of subentries**—Subentries are positioned under an entry and indented two spaces on their own lines.
- **Case**—The .XLOWER and .XUPPER commands control whether index entries are uppercase or lowercase.
- **Merging**—Index entries merge with other entries having identical spelling, spacing, punctuation, and emphasis.

If .XLOWER is in effect, uppercase characters sort before lowercase characters. If .XUPPER is in effect, uppercase and lowercase entries are merged.

**DSR-48     Digital Standard Runoff**  
**Index Formatting**

Entries with different emphasis are sorted in the following order: bolded and underlined, bolded, underlined, no emphasis.

- Sorting .ENTRY entries—Entries without page references are sorted at the beginning of each subindex level.

# Appendix EDT

## EDT Editor

The following appendix lists the specifications of the DCL command EDIT which invokes the EDT editor, the keypad and line editing commands available in EDT, and the nokeypad-editing commands used to redefine keys in EDT.

### EDT.1 EDIT Command

Use the DCL EDIT command to invoke EDT.

#### **EDIT/EDT file-spec**

Invokes the EDT editor.

#### **PARAMETERS**

##### **file-spec**

Specification of the file being edited.

#### **QUALIFIERS**

##### **/COMMAND=file-spec (default)**

##### **/NOCOMMAND**

Determines whether EDT executes a startup command file before the editing session begins and the specification of that command file. The default is /COMMAND=[default-directory]EDTINI.EDT.

##### **/CREATE (default)**

##### **/NOCREATE**

Determines whether EDT creates a new file when the specified input file is not found.

## **EDT-2    EDT Editor**

### **EDIT Command**

#### **/JOURNAL=file-spec (default)** **/NOJOURNAL**

Determines whether EDT keeps a journal file during an editing session and the specification of that journal file. The default is `/JOURNAL=filename.JOU`, where filename is the name of the file being edited.

#### **/OUTPUT=file-spec (default)** **/NOOUTPUT**

Determines whether EDT creates an output file during the editing session and specifies the name of the output file. The default is `/OUTPUT=input-file-spec`, where the file name and type remain unchanged and the version number is incremented by one.

#### **/READ\_ONLY** **/NOREAD\_ONLY (default)**

Determines whether both an output file and a journal file are created. With the default, `/NOREAD_ONLY`, EDT maintains the journal file in case an interruption occurs and creates an output file when the EXIT command is issued. Use the `/READ_ONLY` qualifier when you are merely looking at a file and do not intend to make any changes to it. When you use the `/READ_ONLY` qualifier, enter the QUIT command to exit from EDT (unless you want to specify an output file name with the EXIT command).

#### **/RECOVER** **/NORECOVER (default)**

Determines whether a journal file is executed before the editing session begins. If the name of the journal file is different from that of the input file, you must specify it with the `/JOURNAL` qualifier.

### **EXAMPLES**

**\$ EDIT CHAP.TXT**

Invokes EDT to edit the highest version of the file CHAP.TXT. EDT attempts to read and execute an EDTINI.EDT file in your default directory, creates a journal file, and when you exit from EDT creates an output file named CHAP.TXT with the highest version number.

**\$ EDIT/NOCOMMAND OLDFILE.DAT/OUTPUT=NEWFILE.DAT**

Invokes EDT to edit the highest version of the file OLDFILE.DAT, prevents EDT from attempting to read a startup command file, and creates an output file with the name NEWFILE.DAT.

**\$ EDIT/READ\_ONLY OLDFILE.DAT**

Invokes EDT to read the file OLDFILE.DAT without creating a journal file and without creating an output file.



**\$ EDIT/RECOVER CHAP.TXT**

Invokes EDT to recover edits made during a previously aborted editing session. EDT reads and executes the editing commands in the journal file CHAP.JOU. Once the file has finished processing CHAP.JOU, it returns control to the user.

## **EDT.2   EDT Keypad Editing**

In EDT keypad-editing mode, you use keypad keys and control keys to perform editing functions. To enter change mode for EDT keypad editing you must enter the EDT line-editing command **CHANGE**; text appears on the screen and the keypad-editing keys immediately assume their editing functions.

### **EDT.2.1   Keypad Commands**

Each key on the keypad performs at least one editing command; most perform two. Pressing a key invokes the primary (or upper) function. Pressing the **GOLD** key (labeled PF1) first and then pressing the desired key invokes the alternate (or lower) function. (Do not hold down the **GOLD** key while pressing the other editing key.) In examples, **GOLD** key sequences are shown with the word *GOLD* followed by a backslash (\) and the other editing key. On VT200 series terminals there is a smaller supplemental editing keypad located between the main keyboard and the EDT keypad. See Section EDT.2.3 for more information on the supplemental editing keypad.

# EDT-4    EDT Editor

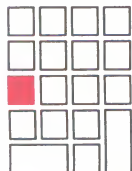
## EDT Keypad Editing

PF1 GOLD	PF2 HELP	PF3 FNDNXT FIND	PF4 DEL L UND L
7 PAGE COMMAND	8 SECT FILL	9 APPEND REPLACE	— DEL W UND W
4 ADVANCE BOTTOM	5 BACKUP TOP	6 CUT PASTE	, DEL C UND C
1 WORD CHNGCASE	2 EOL DEL EOL	3 CHAR SPECINS	ENTER ENTER
0 LINE OPEN LINE	• SELECT RESET	SUBS	

ZK-1688-84

The diagram above each command description represents the EDT keypad shown above.

### ADVANCE



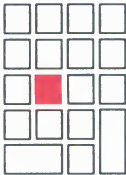
Sets the cursor direction forward, from top to bottom of the buffer, for the following commands: CHAR, WORD, FIND, FNDNXT, CHNGCASE, LINE, EOL, PAGE, SECT, and SUBS, and remains in effect until after you press BACKUP.

#### APPEND



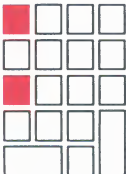
Deletes the select range (see SELECT) and stores it at the end of the PASTE buffer without otherwise modifying the PASTE buffer's original contents.

#### BACKUP



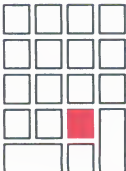
Sets cursor direction backward, from bottom to top of buffer, for the following commands: CHAR, WORD, FIND, FNDNXT, CHNGCASE, LINE, EOL, PAGE, SECT, and SUBS, and remains in effect until you press ADVANCE.

#### BOTTOM



Moves the cursor to the end, or bottom, of the buffer.

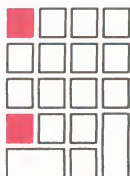
#### CHAR



Moves the cursor one character in the current direction (depending upon whether ADVANCE or BACKUP is set.)

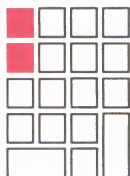
## EDT-6 EDT Editor EDT Keypad Editing

### CHNGCASE



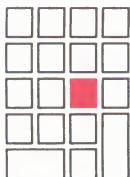
Changes the case (from uppercase to lowercase, or lowercase to uppercase) of all letters in the selected range (see SELECT) or search string (see SET SEARCH). If there is no selected range or the cursor is not positioned on the search string, CHNGCASE changes the case of the current character.

### COMMAND



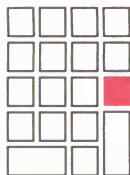
Invokes the "Command:" prompt for entering a line-editing command. Use the ENTER key to process a line-editing command issued at the "Command:" prompt.

### CUT



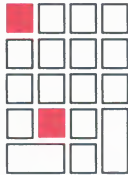
Deletes the selected range (see SELECT) from the current buffer and places the text in the PASTE buffer.

### DEL C



Deletes the character on which the cursor is positioned.

#### DEL EOL



Deletes the text from the cursor to the end of the current line, excluding the line terminator. If the cursor is already at the end of a line, DEL EOL deletes the next line.

#### DEL L



Deletes text from the cursor to the end of the line, including the line terminator. If the cursor is at the beginning of the line, the entire line is deleted, positioning the cursor at the beginning of the next line.

#### DEL W



Deletes the text from the cursor to the first character of the next word. The line terminator (EOL) is treated as a word by the DEL W command.

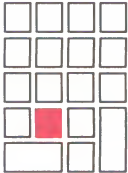
#### ENTER



## EDT-8 EDT Editor EDT Keypad Editing

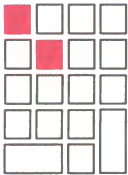
Enters the response to a "Search for:" or "Command:" prompt, or completes the processing of the key definition operation.

### EOL



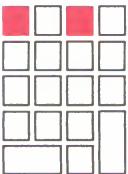
Moves the cursor to the end of the current line or the previous line (depending upon whether ADVANCE or BACKUP is set). If the cursor is already at the end of the line, EOL moves it to the end of the next or previous line.

### FILL



Formats a select range of text by filling each line with as many whole words as possible within the defined line width. (The SET WRAP command defines line width.)

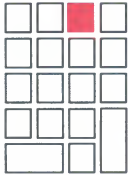
### FIND



Elicits the "Search for:" prompt as the first step in the FIND operation. The command sequence is: FIND (type the search string after the prompt) and ENTER (or ADVANCE or BACKUP). If the string is found, the cursor moves to the first character in the string; otherwise, the cursor remains in place and the message "String was not found" appears. The default search string is the line terminator (EOL).

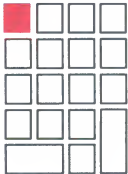


#### FNDNXT



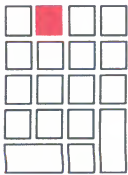
Moves the cursor to the first character of the next occurrence of the search string specified in the FIND command. If there is no further occurrence of the string, the cursor remains in place and the message "String was not found" appears.

#### GOLD



When pressed before another keypad key, specifies that key's alternate function. When pressed before a number and another keypad command, GOLD causes the command to be performed the number of times specified by the number. When used with SPECINS, inserts a character from the DEC Multinational Character set (see SPECINS command). GOLD can be used to define GOLD/keyboard key sequences (see CTRL/K).

#### HELP



Displays a diagram of keypad keys. When one of the keys is pressed after HELP, information about that key is displayed. This function has no effect on the text you are editing. Press the spacebar to return to keypad editing.

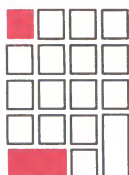
**EDT-10    EDT Editor**  
**EDT Keypad Editing**

**LINE**



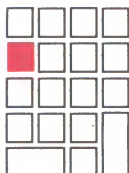
Moves the cursor to the beginning of the next or previous line (depending upon whether ADVANCE or BACKUP is set).

**OPEN LINE**



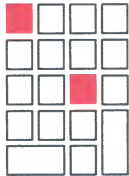
Inserts a line terminator at the current cursor position.

**PAGE**



Moves the cursor to the next or previous page boundary (depending upon whether ADVANCE or BACKUP is set). The page entity defaults to the text between form feeds and can be defined with the SET ENTITY command.

**PASTE**



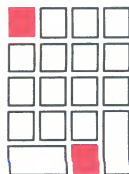
Inserts the contents of the PASTE buffer (the text last affected by the CUT or APPEND command) at the cursor's current position, positioning the cursor at the end of the inserted text.

**REPLACE**



Deletes the selected range and replaces it with the contents of the PASTE buffer (the text last affected by the CUT or APPEND command). The command sequence is: store the new text in the PASTE buffer with SELECT and CUT; locate the old text with FIND and ENTER; mark a selected range of the text to be replaced; and press REPLACE. If a range of text to be replaced is not selected, the text of the search string will be replaced.

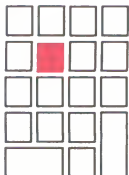
**RESET**



Cancels a selected range, and sets EDT's current direction to ADVANCE.

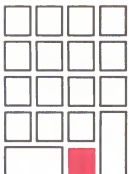
**EDT-12    EDT Editor**  
**EDT Keypad Editing**

**SECT**



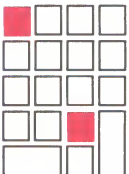
Moves the cursor 16 lines in the current direction (depending upon whether ADVANCE or BACKUP is set).

**SELECT**



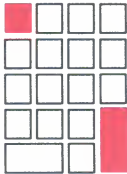
Marks the current cursor position as the beginning of a selected range. The selected range consists of all the text between this marked position and the cursor's subsequent position at the other end of the selected text.

**SPECINS**



Inserts a character (using its decimal value) from the DEC Multinational Character set (see Appendix CHAR). The command sequence is: GOLD, the decimal value of the character, GOLD, and SPECINS.

#### SUBS



Deletes the selected range, replaces it with the contents of the PASTE buffer, and moves the cursor (in the direction set by ADVANCE or BACKUP) to the next occurrence of the search string. The command sequence is: store the new text in the PASTE buffer with SELECT and CUT; locate the old text with FIND and ENTER; replace the old text with the new text and find the next occurrence of the search string using SUBS. If the string is not found, the cursor remains in place and the message "String was not found" appears. If the string is found, you can replace any subsequent occurrences of the search string by entering SUBS.

#### TOP



Moves the cursor to the beginning, or top, of the buffer.

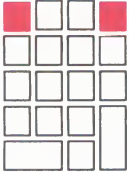
#### UND C



Inserts at the cursor's current position the character most recently deleted with DEL C or DELETE.

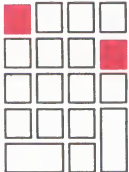
**EDT-14    EDT Editor**  
**EDT Keypad Editing**

**UND L**



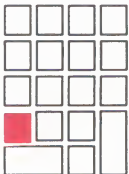
Inserts at the cursor's current position the line of text most recently deleted with DEL L, DEL EOL, or CTRL/U.

**UND W**



Inserts at the cursor's current position the word most recently deleted with DEL W or LINEFEED.

**WORD**



Moves the cursor one WORD in the current direction (depending upon whether ADVANCE or BACKUP is set). You can define the WORD entity with the SET ENTITY command.

You can use the following directional keys:

**DOWN ARROW**



Moves the cursor to the character in the line directly below. If the line of text below does not extend as far as the cursor's position, DOWN ARROW places the cursor on the last character (EOL) in the line below.



#### **LEFT ARROW**



Moves the cursor one character to the left. If the cursor is at the left margin, LEFT ARROW moves the cursor to the last character (EOL) in the previous line.

#### **RIGHT ARROW**



Moves the cursor one character to the right. If the cursor is at the end of the line (EOL), RIGHT ARROW moves it to the leftmost character in the next line.

#### **UP ARROW**



Moves the cursor to the character in the line directly above. If the line of text above does not extend as far as the current cursor position, UP ARROW places the cursor at the last character (EOL) in the line above.

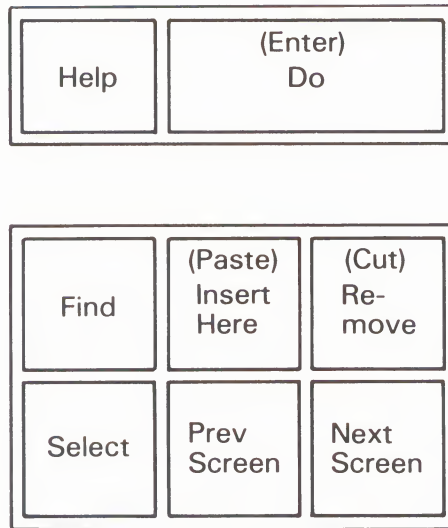
## **EDT.2.2 Keyboard Keys**

You can use the following keyboard keys to supplement the keypad:

- F12 (the BACKSPACE key on VT100 series terminals)—Moves the cursor to the beginning of the current line. If the cursor is already at the beginning of a line, F12 (BACKSPACE) moves it to the beginning of the previous line.
- The Delete ( **<X>** ) key (DELETE on VT100 series terminals)—Deletes the character to the left of the cursor.
- F13 (the LINEFEED key on VT100 series terminals)—Deletes the text from the cursor back to the beginning of the word. If the cursor is on the first character in a word, F13 (LINEFEED) deletes the previous word.
- RETURN—Inserts a line terminator at the current cursor position.
- TAB—Moves the text to the next tab stop.

### **EDT.2.3 VT200 Supplemental Editing Keypad**

The VT200 series terminal has a supplemental editing keypad, which is illustrated in the figure below:



ZK-1677-84

The VT200 supplemental editing keypad keys perform the same functions as some of the EDT keypad keys, and are described below:

- **DO**—Enters the response to a “Search for:” or “Command:” prompt or completes the processing of line-editing commands. (Performs the same function as the ENTER keypad key.)
- **FIND**—Elicits the “Search for:” prompt as the first step in the FIND operation. Type the search string after the prompt and then press either the DO or ENTER key to process the search. (Performs the same function as the FIND keypad key.)
- **HELP**—Displays a diagram of EDT keypad keys. When one of the keys is pressed after HELP, information about that key is displayed. This function has no effect on the text you are editing. (Performs the same function as the HELP keypad key.)
- **INSERT HERE**—Inserts the contents of the PASTE buffer at the cursor’s current position. (Performs the same function as the PASTE keypad key.)

- **NEXT SCREEN**—Moves the cursor forward 16 lines. (Performs the same function as the **ADVANCE SECT** keypad sequence.)
- **PREV SCREEN**—Moves the cursor backward 16 lines. (Performs the same function as the **BACKUP SECT** keypad sequence.)
- **REMOVE**—Deletes the select range (see **SELECT**) from the current buffer and places the text in the **PASTE** buffer. (Performs the same function as the **CUT** keypad key.)
- **SELECT**—Marks the current cursor position as the beginning of the select range. The select range consists of all the text between this marked position and the cursor's subsequent position at the end of the text being selected. (Performs the same function as the **SELECT** keypad key.)

## **EDT.2.4 Control Keys**

You can use the following control keys in EDT keypad editing.

### **CTRL/A**

Establishes the current cursor position and resets the indentation level count to be the quotient of the cursor position divided by the **SET TAB** value. The cursor position must be a multiple of the **SET TAB** value (a valid tab stop); otherwise EDT returns the message "Could not align tabs with cursor" and no action is taken. (**GOLD + A** also performs this function.)

### **CTRL/C**

Aborts the currently executing EDT command.

### **CTRL/D**

Decreases the **TAB** indentation level count one tab setting. (**GOLD + D** also performs this function.)

### **CTRL/E**

Increments the **TAB** indentation level count by one. (**GOLD + E** also performs this function.)

## **CTRL/H**

Like BACKSPACE, moves the cursor to the beginning of the line (or to the beginning of the preceding line if the cursor is already at the beginning of the line).

## **CTRL/I**

Like TAB, moves a line of text to the next tab stop.

## **CTRL/J**

Like LINEFEED, deletes backward from the cursor to the beginning of a word. If the cursor is on the first character in a word, CTRL/J deletes the previous word.

## **CTRL/K**

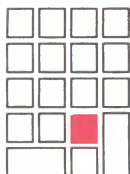
Redefines keypad, control keypad, and gold keypad keys for the current terminal session. When you press CTRL/K, the message "Press the key you wish to define" appears. After you have responded, the message "Now enter the definition terminated by ENTER" appears. Enter the key definition, either by pressing one or more EDT keypad keys or by typing EDT nokeypad editing commands (or a combination of pressing keypad keys and typing nokeypad commands). The guidelines for valid definitions are identical to the DEFINE KEY command, except that you do not need to include the delimiting quotation marks. Terminate the definition by typing a period and pressing the ENTER key.

The following key definition redefines CHAR to transpose the two characters to the left of the cursor. It moves the cursor two characters to the left ( $\leftarrow \leftarrow$ ); deletes the character at the new cursor position (DEL C); moves the cursor one character to the right ( $\rightarrow$ ); restores the deleted character (GOLD and UND C); and then moves the cursor back to its previous position ( $\rightarrow$ ).

**CTRL/K**

Press the key you wish to define:

**CHAR**



Now enter the definition terminated by ENTER

LEFT ARROW



LEFT ARROW



DEL C



RIGHT ARROW



UND C



## **EDT-20    EDT Editor**

### **EDT Keypad Editing**

#### **RIGHT ARROW**



To complete the key definition, type a period on the main keyboard and then press the ENTER key.

#### **CTRL/L**

Inserts a form feed character.

#### **CTRL/M**

Like RETURN, inserts a carriage return (**CR**) into your text.

#### **CTRL/R**

Clears and refreshes the screen, removing extraneous characters and restoring the previous display.

#### **CTRL/T**

If you have previously specified a SET TAB value, CTRL/T indents whole lines in the select range one tab stop to the right. (GOLD + T also performs this function.)

**NOTE:** To allow CTRL/T to work correctly in EDT, you must first disable the DCL function of CTRL/T. By default, DCL displays process statistics when you enter CTRL/T. To disable the DCL function, enter the DCL command SET NOCONTROL=T at the dollar-sign prompt.

#### **CTRL/U**

Deletes from the cursor to the beginning of the line. If the cursor is positioned at the beginning of the line, CTRL/U deletes the previous line.



## **CTRL/W**

Clears and refreshes the screen, removing extraneous characters and restoring the previous display.

## **CTRL/Z**

Changes editing mode from keypad editing to line editing. (GOLD + Z also performs this function.)

## **EDT.3 Line Editing**

Some line-editing commands in EDT require range specifications.

### **EDT.3.1 Range Specifications**

You use the following range specifications as parameters for EDT line-editing commands.

<b>Range</b>	<b>Description</b>
.	The current line. (Example: *DELETE .)
line number	The line indicated by number. Only decimal integers are legal. (Example: *DELETE 22)
'string' or "string"	The next line(s) containing the specified string. The default command, when no other command is specified, is TYPE. (Example: *DELETE 'EDT')
buffer-name	The specified buffer (that may contain a line range). Buffer names must begin with a letter or underscore but may contain numerics. (Example: *DELETE =BUFA 22:33)
BEGIN	The first line in the current buffer. (Example: *DELETE BEGIN)
END	The end of the buffer (marked by the [EOB] symbol by default). (Example: *COPY BEGIN TO END)
LAST	The last line in the previous buffer where the cursor was positioned. The cursor moves to the beginning of this line. (Example: *COPY BEGIN TO LAST)
BEFORE	All lines in the buffer that precede the current line. (Example: *DELETE BEFORE)
REST	The current line plus all lines after it in the buffer. (Example: *DELETE REST)

## EDT-22    EDT Editor

### Line Editing

Range	Description
WHOLE	The entire buffer. (Example: *DELETE WHOLE)
ALL 'string'	All lines containing the specified string. (Example: *DELETE ALL 'EDT')
[range] AND [range]	The specified single lines. (Example: *DELETE 22 AND 33)
[range] , [range]	The specified single lines. (Example: *DELETE 22,33,55)
[range] THRU [range]	The set of lines between the specified ranges. The default for either range is the current line. (Example: *DELETE 22 THRU 33)
[range] : [range]	The set of lines between the specified ranges. The default for either range is the current line. (Example: *DELETE 22:33)
[range] FOR n	Range and the next n lines (where range is a single line that defaults to the current line). (Example: *DELETE 22 FOR 10)
[range] # n	Range and the next n lines (where range is a single line that defaults to the current line). (Example: *DELETE 22 # 10)
[range] + n	The line that is n lines after the range. Range defaults to the current line and n to 1. (Example: *DELETE BEGIN + 3)
[range] - n	The line that is n lines before the range. Range defaults to the current line and n to 1. (Example: *DELETE . - 10)

### EDT.3.2 Line-Editing Commands

Enter EDT line-editing commands after the asterisk prompt. (To obtain the asterisk prompt from EDT keypad-editing mode, enter CTRL/Z.)

#### CHANGE [range]

Changes EDT editing mode from line to either keypad or nokeypad editing (depending on whether you have used the SET KEYPAD or SET NOKEYPAD command). The default is keypad editing. The range specifies the cursor position when you enter keypad or nokeypad editing mode; the range defaults to the cursor's current position.

#### PARAMETERS

##### range

The line at which the cursor is positioned when you enter EDT keypad or nokeypad editing mode. If no range is specified, the range defaults to the cursor's current position.

#### EXAMPLE

\*CHANGE END

Changes from EDT line editing to keypad editing, positioning the cursor at the end of the buffer.

### **CLEAR** buffer-name

Deletes the contents of and removes the specified buffer.

#### PARAMETERS

##### **buffer-name**

The name of the buffer to be cleared. The buffer name must be specified, even if it is the current buffer. The contents of the MAIN and PASTE buffers can be deleted, but the buffers themselves cannot be removed.

#### EXAMPLE

\*CLEAR PASTE

Deletes the contents of the PASTE buffer.

### **COPY** [range-1] TO [range-2]

Inserts the text specified by *range-1* immediately before the line specified by *range-2*. The original text remains intact and the cursor is positioned at the end of the inserted text. You can specify the buffer that contains the range by preceding the range with =buffer name.

#### PARAMETERS

##### **range-1**

The line or lines and/or buffer name containing the text to be copied. If *range-1* is omitted, EDT copies the current line. If a buffer name is specified with no range, the entire contents of the buffer are copied.

##### **range-2**

The line and/or buffer name to be preceded by the copied text. If *range-2* is omitted, EDT inserts a copy of the text preceding the current line. If a buffer name is specified with no range, the text is inserted at the top of the specified buffer.

## EDT-24    EDT Editor

### Line Editing

#### QUALIFIERS

##### **/DUPLICATE:n**

Performs the COPY command n times.

##### **/QUERY**

Prompts for verification of each line as it is copied. Possible responses are:

Y (yes)	Copy this text
N (no)	Do not copy this line
A (all)	Complete the remaining copy operations without further query
Q (quit)	Do not attempt any further copy operations

#### EXAMPLES

**\*COPY 5:10 TO END**

Duplicates lines 5 through 10 at the end of the current buffer.

**\*COPY 25:30 TO END/QUERY/DUPLICATE:3**

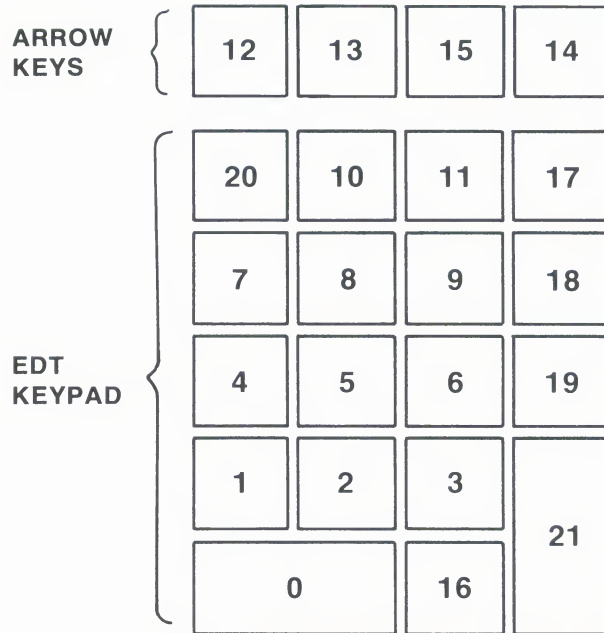
Duplicates lines 25 through 30 at the end of the current buffer, prompting for verification of each line before it is copied. The confirmed lines are copied in their original order three times.

**\*COPY =MAIN TO =BUF1/QUERY**

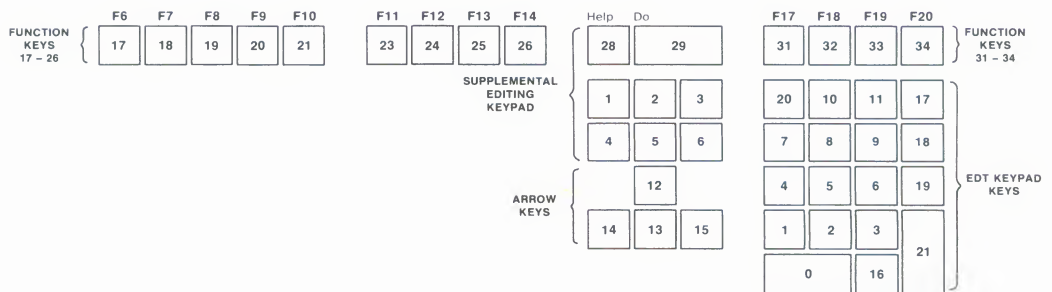
Duplicates the contents of the MAIN buffer (MAIN) in the buffer named BUF1, prompting for verification before each line is copied.

#### **DEFINE KEY** parameter AS 'string'

Defines or redefines the function of keys used in EDT keypad editing by assigning EDT nokeypad-editing commands to keypad, control, or function keys. (See Section EDT.4.2 for a list of these commands.) You must use EDT's numerical designations for the keypad and function keys you define. The figures below illustrate the key numbers for both the VT200 and VT100 series terminals. (To designate VT200 function keys, you must precede the key number with the word *FUNCTION*.)



ZK-1776-84



*Parameter* consists of the name of the key to be defined, and *string* consists of EDT nokeypad-editing commands enclosed in quotation marks. (Valid types of key sequences are described below.) Use a period (.) at the end of the key definition to have EDT execute the command immediately.

## EDT-26    EDT Editor

### Line Editing

To define a key to invoke a prompt, include a question mark (?), in the string followed by the prompt string in quotation marks. Do not use the same type of quotation marks to enclose the prompt string as you use to surround the key definition. You can enter up to 64 characters in response to the prompt, terminated by the ENTER key. To allow the RETURN key to also be used to terminate a response, insert an asterisk (\*) immediately after the question-mark prompt in the key definition.

#### PARAMETERS

##### **[GOLD]number**

Specifies the number of the keypad key to be defined. The optional *GOLD* specifies the alternate function of the key. (See the preceding figure for an illustration of the correct EDT key numbers.)

##### **CTRL/letter**

Specifies the control/key sequence to be defined. *CONTROL* can be specified either with just a keyboard character or with *GOLD* and a keyboard character (where the correct sequence is: *GOLD CONTROL letter*). The following control keys are used by the operating system to perform special functions; therefore, you should not redefine them:

CTRL/C  
CTRL/O  
CTRL/P  
CTRL/Q  
CTRL/R  
CTRL/S  
CTRL/T  
CTRL/U  
CTRL/W  
CTRL/X  
CTRL/Y  
CTRL/Z

##### **FUNCTION number**

Definable function keys on the VT200 series terminal include keys F17 through F20 on the function key row across the top of the keyboard. To define a function key, type the word *FUNCTION* followed by the function key number.

##### **GOLD character**

Specifies the keyboard key to be assigned a new editing function. The keys 0 through 9, !, %, ', -, and " cannot be redefined. To specify the DELETE key, type the reserved word *DELETE* to define that key.

##### **'string'**

A string can contain multiple nokeypad-editing commands (see Section EDT.4.2).



## EXAMPLES

**\*DEFINE KEY GOLD 1 AS 'CHGCW.'**

Redefines the function of keypad GOLD + 1 to change (CHGC) a word (W) from uppercase to lowercase or from lowercase to uppercase.

**\*DEFINE KEY GOLD CONTROL A AS 'BACK C DC ADV C UNDC.'**

Defines the function of GOLD CTRL/A to transpose the two characters to the left of the cursor by setting the cursor direction backward (BACK), moving the cursor one character (C), deleting one character (DC), setting the cursor in the forward direction (ADV), moving the cursor one character (C), and restoring the first character deleted (UNDC).

**\*DEFINE KEY FUNCTION 19 AS "EXT S/?'REPLACE: '/?' WITH: '/WHOLE."**

Redefines the function of key F19 to substitute one string for another throughout a buffer by extending the line-editing command (SUBSTITUTE) to EDT keypad editing (EXT) and substituting the specified text (entered after the prompt REPLACE:) with the specified text (entered after the prompt WITH:) throughout the buffer (WHOLE).

## DEFINE MACRO macro-name

Defines the macro name as an EDT line-editing command. To use the new command, create a buffer with the same name as the macro and insert the sequence of EDT commands (the macro) that you want to execute whenever you type the macro name after the asterisk prompt. A macro can be saved in an external file and copied into a buffer during your editing session with the INCLUDE command. If you redefine an ED line-editing command, the original command is not available as long as the macro definition is in effect. To delete a macro, use the macro name as the buffer specifier with the CLEAR command. You can nest macros to almost any depth.

## PARAMETERS

### macro-name

The name of the EDT macro containing line-editing commands. The macro name must be the same as the name of the buffer containing the EDT macro.

## EDT-28    EDT Editor

### Line Editing

#### EXAMPLE

```
*DEFINE MACRO HEADING
*FIND=HEADING
INSERT;NAME:
INSERT;DEPT:
INSERT;DATE:
INSERT;SUBJ:
*FIND=MAIN .
```

Defines a macro named HEADING to insert a heading format whenever you enter HEADING as an EDT line-editing command. To place the preceding macro in an EDT startup command file, include the following commands:

```
DEFINE MACRO HEADING
FIND=HEADING
INSERT;INSERT;NAME:
INSERT;INSERT;DEPT:
INSERT;INSERT;DATE:
INSERT;INSERT;SUBJ:
FIND=MAIN .
```

The DEFINE MACRO command is necessary to create the macro definition; the FIND commands are necessary to enter and exit from the buffer containing the macro; and the INSERT command is necessary to enter each command in the macro sequence (hence, the double INSERT commands).

## DELETE [range]

Deletes the lines specified by the range.

### PARAMETERS

#### range

The line or lines specified for deletion. The range defaults to the current line.

### QUALIFIERS

#### /QUERY

Prompts for verification before each line is deleted. Possible responses are:

Y (yes)	Delete this line
N (no)	Do not delete this line
A (all)	Complete the remaining deletions without further query
Q (quit)	Do not make any further deletions

#### EXAMPLE

**\*DELETE 12:56/QUERY**

Deletes lines 12 through 56, prompting for verification.

#### EXIT [file-spec]

Ends an editing session, writing the contents of the MAIN buffer to an output file.

#### PARAMETERS

##### [file-spec]

Defines the file specification of the output file. The default is the highest version of the input file named in the EDIT command line (or the output file named with the DCL command EDIT/OUTPUT=file-spec). If you enter EDT with the DCL command EDIT/READ\_ONLY, the EXIT command must include an output file specification.

#### QUALIFIERS

##### /SEQUENCE[:initial[:increment]]

Assigns sequence numbers, which become part of the file. The initial qualifier defines the starting line number. The increment qualifier defines the increment between sequence numbers. (This qualifier should not be used to number programs since references to line numbers within programs will not be changed.)

##### /SAVE

Saves the journal file, which has the file type JOU and the file name of the input file specified in the EDIT command line.

#### EXAMPLES

**\*EXIT**

Saves the contents of the MAIN buffer in an output file and returns you to DCL command level.

**\*EXIT NAME.NEW/SEQUENCE:5:5/SAVE**

Saves the contents of the MAIN buffer in the output file NAME.NEW. The /SEQUENCE qualifier assigns sequence numbers that become part of the file: the sequence number of the first line in the file is 5, and the following lines are incremented by five. The /SAVE qualifier saves the journal file.

## **FILL [range]**

Reorganizes the text in a select range of lines so that the maximum number of whole words are fitted within the current line width. The default line width for EDT is 80 characters. Use the line-editing command SET WRAP to change the line width for the FILL operation.

### **PARAMETERS**

#### **range**

A line or lines and/or buffer name to be filled. If you do not specify a range or buffer name with the FILL command, EDT attempts to fill the text in the active select range. If no select range is active EDT displays an error message.

### **EXAMPLE**

**\*FILL 10 THRU 15**

Fills lines 10 through 15 to the value of SET WRAP or SET SCREEN.

## **FIND [range]**

Places the cursor at the first line of the specified range in the specified or current buffer, but does not display the line at the bottom of your screen. In line mode, once the specified line has been found, EDT returns the line-mode asterisk (\*) prompt. (Use the TYPE command to display the line.) The FIND command searches either forward or backward for a string in the current buffer. To search backward toward the beginning of the buffer, precede the search string (range) with a minus sign (-). If you specify a buffer that does not exist, EDT creates it. To move the cursor to its previous position in a buffer, include a period (.) immediately after the buffer name. (The FIND command is useful for moving from one buffer to another.)

### **PARAMETERS**

#### **range**

A line and/or buffer name. When a buffer name is specified without a range, EDT positions the cursor at the first character of the buffer. When both a buffer name and range are specified, EDT positions the cursor at the line specified by *range* in the specified buffer. Use a space to separate the buffer name from the range specifier. (If you use a string as a range specifier, EDT performs the search using the defaults: GENERAL, BEGIN, and UNBOUNDED.)

### **EXAMPLES**

**\*FIND =BUFA 45**

Positions the cursor at line 45 in the buffer named BUF1.

**\*FIND=MAIN.**

Returns the cursor to its previous position in the MAIN buffer.

## **HELP [topic [subtopic]]**

Displays a list of EDT topics on which you can get information.

### **PARAMETERS**

#### **topic**

The topic for which you wish to display information.

#### **subtopic**

The subtopic for which you wish to display information.

### **EXAMPLE**

**\*HELP FILL**

Displays information about the EDT command FILL.

## **INCLUDE file-spec [range]**

Copies the specified file to a text buffer, placing it immediately before the range and positioning the cursor immediately following the inserted text.

### **PARAMETERS**

#### **file-spec**

The name of the file you wish to include in the specified buffer.

#### **range**

A line number or buffer name. If no buffer is specified, the file is added to the current buffer. The range defaults to the current line in the specified buffer.

### **EXAMPLES**

**\*INCLUDE CHAP1.TXT**

Copies the contents of the file named CHAP1.TXT into the current buffer just before the current line.

**\*INCLUDE CHAP1.TXT =BUF1 30**

Copies the contents of the file named CHAP1.TXT into the buffer named BUF1, beginning before line 30.

## **INSERT [range]**

Inserts text before the specified single line range. To insert a full line or more of text, press RETURN after the optional range specifier, followed by lines of text terminated with a CTRL/Z. To insert one line of text, place a semicolon after the range; any text typed after the semicolon is inserted before the range when you press RETURN. (Use only the semicolon insertion in startup command procedures.) If lines are inserted between two lines whose line numbers differ by one or less, EDT will number the new lines using decimal fractions. (In extreme cases EDT may be forced to renumber lines after the last line you insert.)

### **PARAMETERS**

#### **range**

The line and/or buffer before which the text is to be inserted. When a buffer name is specified, EDT moves to that buffer and remains there after the insertion has been made. The buffer name defaults to the current buffer and the range defaults to the current line.

### **EXAMPLES**

**\*INSERT 25**

Inserts lines of text before line 25.

**\*INSERT =BUF1 30;ENTER YOUR BADGE NUMBER**

Inserts "ENTER YOUR BADGE NUMBER" before line 30 in the buffer named BUF1.

## **MOVE [range-1] TO [range-2]**

Deletes the lines specified in *range-1* and inserts them before the line specified in *range-2* (which then becomes the current line). (COPY inserts the text in both places.)

### **PARAMETERS**

#### **range-1**

The line or lines and/or buffer name containing the text to be moved. If *range-1* is omitted, EDT moves the current line. If a buffer name is specified with no range, the entire contents of the buffer are moved.

#### **range-2**

The line and/or buffer name to be preceded by the moved text. If *range-2* is omitted, EDT inserts the text just before the current line. If a buffer name is specified with no range, the text is inserted at the top of the specified buffer.



## QUALIFIERS

### /QUERY

Prompts for verification of each line before it is moved. Possible responses are:

Y (yes)	Move this text
N (no)	Do not move this text
A (all)	Complete the remaining move operations without further query
Q (quit)	Do not move any more lines of text

### EXAMPLE

**\*MOVE TO 65**

Deletes the current line and inserts it before line 65.

## PRINT file-spec [range]

Copies the specified range to a printable file; that is, PRINT inserts a form feed and two blank lines every 55 lines, and line numbers become part of the new file's text. The file is submitted to SYS\$PRINT and is printed.

### PARAMETERS

#### file-spec

The specification of the file to be printed.

#### range

The line or lines and/or buffer name containing the text to be printed. The range defaults to the entire current buffer.

### EXAMPLE

**\*PRINT NEWFILE.DAT 22:30**

Copies lines 22 through 30 to the printable file named NEWFILE.DAT.

## QUIT

Terminates EDT without saving the edits made during the session.

## QUALIFIERS

### /SAVE

Saves the journal file, which has the file type JOU and the file name of the input file (unless you used /JOURNAL in the EDIT command line).

## REPLACE [range]

Deletes the specified range and inserts new text in place of the deleted range. EDT rennumbers the inserted text, using decimal line numbers when the new text is longer than the old. (When the new text consists of fewer lines than the old, the extra line numbers are not used.)

### PARAMETERS

#### range

The line or lines containing the text to be replaced. The range defaults to the current line. To insert a full line or more, press RETURN and type the text, terminating it with CTRL/Z. To insert one line, enter a semicolon after the range; all text typed after the semicolon is inserted in place of the range when you press RETURN.

### EXAMPLE

\*REPLACE 3; I regret to inform you that RETURN

Replaces line 3 with the string "I regret to inform you that"

## RESEQUENCE [range]

Assigns new EDT line numbers to the specified range.

### PARAMETERS

#### range

The consecutive lines or buffer to be renumbered. The range defaults to the entire current buffer.

### QUALIFIERS

#### [/SEQUENCE[:initial[:increment]]]

Assigns line numbers according to the specified starting number *initial* and increment value *increment*. (The defaults for *initial* and *increment* are 1.)

### EXAMPLE

\*RESEQUENCE =BUFA

Resequences the line numbers of the lines in the buffer named BUFA.

## SET [NO]AUTOREPEAT

Enables EDT to prevent keypad keys (including arrow keys) from repeating faster than EDT can update the screen. SET AUTOREPEAT is the default.

#### EXAMPLE

**\*SET NOAUTOREPEAT**

Prevents keypad keys from repeating more rapidly than EDT can update the screen.

### SET CASE [[none]][upper][lower]]

Distinguishes uppercase and lowercase letters when you are reading text at a single-case terminal. No permanent marks are placed in the text itself. The default is SET CASE NONE.

#### PARAMETERS

##### **none**

Causes no provision to be made for artificially distinguishing between uppercase and lowercase letters.

##### **upper**

Causes each uppercase letter in the text to be preceded by an apostrophe (').

##### **lower**

Causes each lowercase letter in the text to be preceded by an apostrophe (').

#### EXAMPLE

**\*SET CASE UPPER**

Assuming you are at a single-case terminal, flags all letters that are uppercase in the original text.

### SET COMMAND file-spec

Specifies an additional EDT startup command file (besides the default EDT startup command file, EDTINI.EDT) when you invoke EDT. The default file type is EDT. If the specified file does not exist, EDT ignores the SET COMMAND and continues to process any remaining commands in the current startup command file. (To bypass the default startup command file, specify a different startup command file in your EDT startup command file.)

#### PARAMETERS

##### **file-spec**

The specification of the startup command file you wish to process.

**EXAMPLE**

**\*SET COMMAND USEREDT.EDT**

As a line in an EDT startup command file, transfers control to another EDT startup command file, USEREDT.EDT.

**SET CURSOR top:bottom**

Determines at which lines scrolling begins: top is the upper limit and bottom is the lower. EDT scrolls the display when the cursor reaches either limit.

**PARAMETERS**

**top**

The number of lines from the top of the screen to the cursor. The allowable limits for top are 0 through 21; the default is top=7.

**bottom**

The number of lines from the cursor to the bottom of the screen. The allowable limits for bottom are 0 through 21; the default is bottom=14.

**EXAMPLE**

**\*SET CURSOR 0:5**

Sets the scrolling region to the first 6 lines on the screen. (If you try to move the cursor beyond either limit, EDT scrolls the display.)

**SET ENTITY entity 'boundary'**

Defines the boundaries for an entity.

**PARAMETERS**

**boundary**

The delimiter values for WORD, SENTENCE, PARAGRAPH, and PAGE. The default boundaries are as follows:

WORD	SENTENCE	PARAGRAPH	PAGE
<LF>	.	<CR> <CR>	<FF>
<VT>	?		
<FF>	!		
<CR>			

### **entity**

The WORD, SENTENCE, PARAGRAPH, or PAGE for which you wish to set boundaries.

#### **EXAMPLE**

```
*SET ENTITY WORD '<LF><VT><FF><CR>. ? ! ; : '
```

Defines the following as delimiters for the WORD entity: line feeds, vertical tabs, form feeds, carriage returns, periods, question marks, exclamation points, semicolons, colons, and individual spaces. All commands using the WORD entity (such as DEL W) work on a unit of characters up to these delimiters.

### **SET [NO]FNF**

Suppresses the “Input file does not exist” message that appears by default when you use EDT to create a new file.

### **SET HELP file-spec**

Enables you to access HELP files other than the default EDT help files (found in SYS\$HELP:EDTHELP.HLB).

#### **PARAMETERS**

##### **file-spec**

The specification of the HELP file you wish to access.

#### **EXAMPLE**

```
*SET HELP DISK1:USEREDTHELP.HLB
```

Enables you to access the help file USEREDTHELP.HLB on device DISK1.

### **SET [NO]KEYPAD**

Sets either EDT keypad or nokeypad editing as the default editing mode. The default setting is KEYPAD.

#### **EXAMPLE**

```
*SET NOKEYPAD
```

Nokeypad becomes the editing mode when you give the CHANGE command.

## **SET LINES** number

Sets the number of lines that EDT displays on your screen while keypad editing. (This command is useful in reducing the time required to refresh the screen image when editing at low baud rates.)

### **PARAMETERS**

#### **number**

The number of lines can be set from 1 to 22; the default setting is 22 lines.

### **EXAMPLE**

**\*SET LINES 10**

Causes EDT to display 10 lines of text on the screen while keypad editing.

## **SET MODE CHANGE or LINE**

Placed in an EDTINI.EDT startup command file, SET MODE specifies either keypad (CHANGE) or line (LINE) mode as the initial editing mode. The SET MODE command should be used only in an EDT startup command file. Change mode is set by default.

### **EXAMPLE**

**SET MODE CHANGE**

Causes EDT to enter change mode for keypad editing after the startup command file has executed.

## **SET [NO]NUMBERS**

Determines whether or not EDT displays line numbers in line-editing mode. The default setting is NUMBERS.

### **EXAMPLE**

**\*SET NONUMBERS**

Suppresses the line numbers normally displayed in line-editing mode.

## **SET [NO]QUIET**

Suppresses the ringing of the bell when a message occurs in keypad-editing mode. The default setting is NOQUIET.



EXAMPLE

**\*SET QUIET**

Causes EDT to suppress the bell when a message occurs in keypad-editing mode.

**SET [NO]REPEAT**

Controls the use of the GOLD repeat feature, which allows you to repeat EDT keypad functions by pressing GOLD and keyboard digits; and the SPECINS keypad function, which enables you to insert any character from the DEC Multinational Character set into your text by using its decimal equivalent value. SET REPEAT is the default.

EXAMPLE

**\*SET NOREPEAT**

Disallows the use of the GOLD repeat feature and the SPECINS keypad function.

**SET SCREEN width**

Sets the maximum length of line that EDT displays.

PARAMETERS

**width**

The length of the line to be displayed; the default line width set by VAX/VMS is 80 characters. In EDT keypad editing, when you insert more characters than the length set for character editing, EDT displays a solid diamond at the line's end. In line-editing, EDT displays the overflow characters on succeeding lines.

EXAMPLE

**\*SET SCREEN 132**

Sets the maximum length of line displayed to 132.

**SET SEARCH parameter**

Controls string searches with the FIND command.

## EDT-40    EDT Editor

### Line Editing

#### PARAMETERS

##### parameter

The characteristics that determine how a search is conducted. Possible parameters are:

GENERAL (default) or EXACT

Determines whether or not EDT disregards the case of alphabetic characters

BEGIN (default) or END

Determines whether EDT positions the cursor at the beginning or end of the found string

UNBOUNDED (default) or BOUNDED

Determines whether EDT searches the entire buffer or stops the search when it reaches a page boundary marker

#### EXAMPLE

```
*SET SEARCH EXACT
*SET SEARCH BOUNDED
*SET SEARCH END
*TYPE 'Mark'
```

Causes EDT to search between the current cursor position and the next page delimiter for the first occurrence of the exact string "Mark" (that is, the first character of the string must be in uppercase and the rest in lowercase). When the string is found, the cursor will be positioned after the string.

#### SET [NO]SUMMARY

Controls whether or not EDT prints a summary (the complete file specification and the number of lines in the file) when you exit from EDT and when you issue the WRITE command. The default is SET SUMMARY.

#### EXAMPLE

```
*SET NOSUMMARY
```

Suppresses the summary information generated when you exit from EDT.

#### SET TAB n or SET NOTAB

Controls the number of columns to which the first tab stop is set. Only the initial tab stop is affected; the remaining tab stops, multiples of 8, are unchanged. SET NOTAB, which is the default, sets the first tab stop at 8 columns from the left margin. SET TAB n sets the first tab stop at n columns from the left margin.

The numbers below indicate column numbers.

#### EXAMPLE

123456789

\*SET TAB 5

\*CHANGE

It is with great pleasure that I announce the winner RETURN

<TAB>

of the Steeple Creek sales award: RETURN

<TAB><TAB>

Roberta Brown of Lancaster County)

By default, the first tab stop is set to 8 columns. The SET TAB 5 command sets the first tab stop to 5 columns; the second tab stop is still set to 8 columns—the next multiple of 8 after the SET TAB value. (The third tab stop is at 16 columns, the fourth at 24, and so on in multiples of 8.)

## SET TERMINAL

Resets the following terminal characteristics:

Terminal type (HCPY, VT100)

Scrolling regions (SCROLL, NOSCROLL)

Additional character display (EIGHTBIT,NOEIGHTBIT)

Enhanced screen editing (EDIT,NOEDIT)

Do not specify SET TERMINAL from keypad-editing mode.

## SET TEXT

Specifies a string to be displayed at the end of a page (PAGE) or buffer (END) by controlling the display of the <FF> character. SET TEXT PAGE displays your string for every form feed in the buffer; SET TEXT END displays your string at the end of the buffer. Specify the string in quotation marks. The default strings displayed are: PAGE= <FF> and END=[EOB].

#### EXAMPLES

\*SET TEXT PAGE "<PAGE BREAK>"

Displays the string <PAGE BREAK> for each form-feed character.

\*SET TEXT END "[END OF BUFFER ADD]"

Displays the string [END OF BUFFER ADD] at the end of the current buffer (replacing the [EOB] symbol).

## **SET [NO]TRUNCATE**

Determines whether the display of lines that are longer than the SET SCREEN value will be truncated or wrapped to the next line. The default setting is TRUNCATE.

### **EXAMPLE**

**\*SET NOTRUNCATE**

Causes the display of lines that are longer than the screen width to be wrapped to the next line.

## **SET [NO]VERIFY**

Displays the commands in startup command files and macros as they are executed. The default setting is NOVERIFY.

### **EXAMPLE**

**\*SET VERIFY**

Causes EDT to display the commands in startup command files and macros as they are executed.

## **SET WORD [NO]DELIMITER**

Controls whether or not EDT treats word boundaries (defined with SET ENTITY) as words. By default, EDT considers all word boundaries except spaces to be words when it is deleting words or moving the cursor by word. Specify SET WORD NODELIMITER to change the default.

### **EXAMPLE**

**\*SET WORD NODELIMITER**

Causes EDT to use word boundaries to determine where words begin and end but not to treat these delimiters as separate words.

## **SET WRAP n or NOWRAP**

In keypad editing, specifies a right margin where, when the cursor position exceeds the value of *n*, EDT wraps the full word to the next line. This command also sets the right margin for the FILL command, which fills each line in the select range to the word delimiter nearest the limit *n*. The default setting is NOWRAP.

#### EXAMPLE

**\*SET WRAP 60**

Causes EDT to wrap words exceeding 60 characters in length to the next line and sets the right margin to 60 for the FILL command.

### SHOW AUTOREPEAT

Displays the current setting of the SET AUTOREPEAT command.

#### EXAMPLE

**\*SHOW AUTOREPEAT**

noautorepeat

Indicates that the SET NOAUTOREPEAT command is in effect.

### SHOW BUFFER

Lists the buffers in use during the current editing session and the number of lines of text in each buffer. The MAIN and PASTE buffers are always displayed in response to this command, even when empty. The current buffer is marked by an equal sign before the buffer's name. An asterisk following the MAIN buffer line count indicates that the input file has not yet been read to the end of the file, and therefore the line count is not accurate.

#### EXAMPLE

**\*SHOW BUFFER**

```
MAIN      396  LINES
=BUF1      10  LINES
PASTE      4   LINES
```

Indicates that the buffers MAIN, BUF1, and PASTE are being used in the current editing session, and that BUF1 is the current buffer.

### SHOW CASE

Indicates whether uppercase or lowercase, or neither, has been established by the SET CASE command.

#### EXAMPLE

**\*SHOW CASE**

None

Indicates that the SET CASE NONE command is in effect.

## **SHOW COMMAND**

Displays the name of the current startup command file when SHOW COMMAND is included in a startup command file.

### **EXAMPLE**

**\*SHOW COMMAND**

\*

Indicates that there is no additional startup command file besides the default EDTINI.EDT.

## **SHOW CURSOR**

Shows the cursor's current scrolling range. EDT responds with "top:bottom," which are integers indicating the top and bottom lines of the scrolling range. The SET CURSOR command sets the scrolling range.

### **EXAMPLE**

**\*SHOW CURSOR**

5:5

Indicates that EDT will scroll at the upper and lower limits of 5.

## **SHOW ENTITY WORD or SENTENCE or PAGE or PARAGRAPH**

Shows the current boundaries for the specified entity. The SET ENTITY command defines these entities.

### **EXAMPLE**

**\*SHOW ENTITY PAGE**

<FF>

Indicates that the current boundary for PAGE is a form feed ( <FF> ).

## **SHOW FILES**

Displays the input file and output file for the current EDT session.

### **EXAMPLE**

**\*SHOW FILES**

Input File: TASKS.DAT

Output File: TASKS.DAT

Indicates that the name of the output file will be the same as the input file with the version number incremented by one. Use the /OUTPUT=file-spec qualifier of the DCL command EDIT to set the name of the output file.



## **SHOW FNF**

Displays the current setting of the SET FNF command.

EXAMPLE

```
*SHOW FNF  
fnf
```

Indicates that the SET FNF command is in effect.

## **SHOW HELP**

Displays the current EDT HELP file. The default is SYS\$HELP:EDTHELP:HLB.

EXAMPLE

```
*SHOW HELP  
SYS$HELP:EDTHELP:HLB
```

Indicates that the current EDT HELP file is set to the default.

## **SHOW KEY**

Displays the definition of the specified keypad or keyboard key. To refer to a GOLD or control key sequence, type GOLD then the key or CONTROL plus the key. (The DEFINE KEY command and CTRL/K define keys.)

EXAMPLE

```
*SHOW KEY GOLD P  
TOP
```

Shows the current definition for the GOLD + P key sequence set with the SET KEY command.

## **SHOW KEYPAD**

Displays the current keypad-editing mode: KEYPAD or NOKEYPAD.

EXAMPLE

```
*SHOW KEYPAD  
keypad
```

Indicates that the current keypad-editing mode is keypad.

## **SHOW LINES**

Shows the number of lines displayed on the screen at one time (set with SET LINES).

EXAMPLE

**\*SHOW LINES**

22

Indicates that the current setting of the SET LINES command is 22.

## **SHOW MODE**

Displays the last SET MODE command setting (which is set in an EDT startup command file). (Does not display modes set by the CHANGE, CTRL/Z, or EX commands.)

EXAMPLE

**\*SHOW MODE**

Change

Indicates that the last command mode setting was SET MODE CHANGE.

## **SHOW NUMBERS**

Indicates whether or not line numbers are currently displayed (set with SET NUMBERS).

EXAMPLE

**\*SHOW NUMBERS**

nonumbers

Indicates that the current setting is SET NONUMBERS.

## **SHOW QUIET**

Displays the current setting of the SET QUIET command.

EXAMPLE

**\*SHOW QUIET**

noquiet

Indicates that the current setting is SET NOQUIET.

## SHOW REPEAT

Displays whether or not you can use the GOLD repeat feature and the SPECINS function in keypad-editing mode (set with the SET REPEAT command).

### EXAMPLE

\*SHOW REPEAT

repeat

Indicates that the current setting is SET REPEAT.

## SHOW SCREEN

Shows the current setting for the maximum length of a line EDT displays. (The SET SCREEN command defines line length.)

### EXAMPLE

\*SHOW SCREEN

80

Indicates that the maximum length of line displayed is 80 characters.

## SHOW SEARCH

Shows the current search parameters (set with the SET SEARCH command).

### EXAMPLE

\*SHOW SEARCH

general begin unbounded

Indicates that the current settings are SET SEARCH GENERAL, SET SEARCH BEGIN, and SET SEARCH UNBOUNDED.

## SHOW SUMMARY

Displays the current setting of the SET SUMMARY command.

### EXAMPLE

\*SHOW SUMMARY

summary

Indicates that the current setting is SET SUMMARY.

## **SHOW TAB**

Displays the current setting of SET TAB and the current tab indentation level count.

### **EXAMPLE**

**\*SHOW TAB**

tab size 5; tab level 1

Indicates that the current tab setting is 5 and the tab indentation level count is 1.

## **SHOW TERMINAL**

Displays the current terminal settings.

### **EXAMPLE**

**\*SHOW TERMINAL**

VT100, scroll, noeightbit, edit

Displays the current terminal characteristics.

## **SHOW TEXT PAGE or END**

Indicates what text EDT is currently displaying for the <FF> and [EOB] marks (set with SET TEXT).

### **EXAMPLE**

**\*SHOW TEXT END**

[THE END]

Indicates that the current end-of-buffer setting is SET TEXT END "[THE END]".

## **SHOW TRUNCATE**

Displays the current setting of the SET TRUNCATE command.

### **EXAMPLE**

**\*SHOW TRUNCATE**

nottruncate

Indicates that the current setting is SET NOTRUNCATE.

## **SHOW VERIFY**

Displays the current setting of SET VERIFY, which controls whether or not the commands in an EDT startup command file or macro are displayed. SET NOVERIFY is the default.

#### EXAMPLE

\*SHOW VERIFY

noverify

Indicates that the current setting is SET NOVERIFY.

### SHOW VERSION

Displays the current version of EDT.

#### EXAMPLE

\*SHOW VERSION

V3.00-17                      COPYRIGHT (C) DIGITAL EQUIPMENT CORPORATION 1980, 1984

Displays the current version of EDT.

### SHOW WORD

Displays the current setting of the SET WORD command.

#### EXAMPLE

\*SHOW WORD

delimiter

Indicates that the current setting is SET WORD DELIMITER.

### SHOW WRAP

Displays the current setting of SET WRAP.

#### EXAMPLE

\*SHOW WRAP

75

Indicates that the current setting is SET WRAP 75.

### SUBSTITUTE \string-1\string-2\[range]

Replaces occurrences of one specified string with another string. A string can be from 0 to 64 characters. Except for the percent sign (%), you can use any nonalphanumeric character that does not appear in either of the strings as a delimiter. You must use the same character as delimiter throughout the command line. (Slashes preceding qualifiers are required.)

## **EDT-50    EDT Editor**

### **Line Editing**

#### **PARAMETERS**

##### **[range]**

EDT replaces all the occurrences of string-1 within the specified range with string-2 and displays the number of substitutions made. At the end of the substitution, the cursor returns to the first line in the specified range. If you do not specify a range, EDT replaces only the first occurrence of the string in the current line.

##### **string-1**

The text string to be replaced.

##### **string-2**

The replacement text string.

#### **QUALIFIERS**

##### **/BRIEF[:n]**

Displays the first n characters of the lines containing string-1. The value of n defaults to 10.

##### **/QUERY**

Prompts for verification before each substitution.

##### **/NOTYPE**

Prevents the display of lines containing substitutions.

#### **EXAMPLES**

**\*SUBSTITUTE\INADVERTANT\INADVERTENT\WHOLE**

Substitutes "inadvertent" for each occurrence of "inadvertant" in the buffer. EDT displays both the number of substitutions made and the lines containing the substitutions.

**\*SUBSTITUTE\*section\*chapter\*20:50/QUERY**

Within the range of lines from 20 through 50, substitutes "chapter" for "section," prompting for verification before each substitution.

##### **[SUBSTITUTE] NEXT [string-1/string-2/]**

Replaces the next occurrence of string-1 with string-2 and positions the cursor at the line where the substitution is made. String-1 defaults to the current search string, and string-2 to the last specified string-2.



## PARAMETERS

**string-1**

The string to be replaced.

**string-2**

The replacement text string.

## EXAMPLE

\*NEXT

If your last substitute command was: "SUB/several/some/", the NEXT (SUBSTITUTE NEXT) command above will find the next occurrence of the word "several" and replace it with "some", displaying the corrected line (so long as you haven't changed the search string in the meantime).

**TAB ADJUST [-]n [=buffer] [range]**

Indents whole lines in the specified range n number of tab stops to create layered text. A minus sign before the n specifier allows you to move text back toward the left margin. In order for the command to work you must first establish a SET TAB value for your editing session.

## PARAMETERS

**=buffer**

The name of the buffer to be indented.

**range**

The line or lines to be indented. If you include a buffer specifier with no range, the entire buffer is indented. When you omit both the buffer and range specifiers, EDT assumes that you have an active select range (see the keypad SELECT command).

**[-]n**

The number of tab stops you wish to indent the text.

## EXAMPLES

\*TAB ADJUST 1 5

Indents line number 5 one tab stop.

\*TAB ADJUST -4 25

Moves (indented) line number 25 four tab stops to the left.

## **TYPE [range][:n]**

Displays a specified range of lines and positions the cursor at the first line of the specified range.

### **PARAMETERS**

#### **range**

The range of lines to be displayed. The range defaults to the current line.

### **QUALIFIERS**

#### **/BRIEF[:n]**

Displays the first n characters of the lines contained in the specified range. The value of n defaults to 10.

#### **/STAY**

Prevents the cursor from moving to the first line of the range.

### **EXAMPLES**

#### **\*TYPE WHOLE**

Displays the entire contents of the current buffer.

#### **\*TYPE 27**

Displays line 27.

#### **\*TYPE 40 THRU END**

Displays all lines from line 40 through the last line of the buffer.

#### **\*TYPE 40:END all "news"**

Displays all lines containing the string "news" that occur between line 40 and the end of the buffer.

## **WRITE filename [range]**

Copies the specified range of text to the specified file.

### **PARAMETERS**

#### **range**

The line or lines of text to be written. The range defaults to the entire current buffer.

### **QUALIFIERS**

#### **/SEQUENCE[:initial[:increment]]**

Assigns sequential line numbers to the file. The initial qualifier defines the starting line number; the increment qualifier defines the increment between line numbers.

## EXAMPLES

**\*WRITE NEWCHAP.TXT**

Copies the entire contents of the current buffer to the file NEWCHAP.TXT. EDT displays the specification of the new file and the number of lines written to the file.

**\*WRITE NEWCHAP.TXT 10:90**

Copies lines 10 through 90 of the current buffer to the file NEWCHAP.TXT. EDT displays the specification of the new file and the number of lines written to the file.

**\*WRITE NEWCHAP.TXT =BUF1/SEQUENCE:10:10**

Copies the contents of BUF1 to the file NEWCHAP.TXT and rennumbers the lines (beginning with line 10 and incrementing by 10). EDT displays the specification of the new file and the number of lines written to the file.

## EDT.4 Nokeypad Editing

You can redefine keypad keys and control keys by using the DEFINE KEY line-editing command with EDT nokeypad-editing commands as the definition. EDT nokeypad commands cannot contain spaces; for example, to delete two paragraphs and put the text in a buffer named EXTRA, type CUT2PAR=EXTRA. You can put several EDT nokeypad commands on the same line; spaces between commands are allowed but not required. You can repeat a series of commands by preceding the commands with the repeat count and enclosing them in parentheses 3(V D+EL).

### EDT.4.1 Text Entities

You can use the following text entities as parameters for EDT nokeypad-editing commands. The parameters are presented in uppercase, followed by their functions in parentheses.

#### PARAMETERS

##### **C (character)**

Single alphabetic, numeric, or special character.

##### **W (word)**

User-defined entity. The default boundaries are spaces, line terminators, tabs, and form feeds. You define the word boundaries with the SET ENTITY command.

##### **BW (beginning of word)**

The string of characters from the cursor to the beginning of the word; when the cursor is at the beginning of a word, BW is the previous word.

**EDT-54    EDT Editor**  
**Nokeypad Editing**

**EW (end of word)**

The string of characters from the cursor position to the end of the word, including the character at the cursor position.

**L (line)**

Single line of text, where L is a string of characters between line terminators.

**BL (beginning of line)**

The string of characters from the cursor position to the beginning of the line. When the cursor is at the beginning of a line, BL is the previous line.

**EL (end of line)**

The string of characters from the cursor position to the end of the line, including the character at the cursor position.

**NL (next line)**

The string of characters from the cursor position to the beginning of the next line.

**V (vertical)**

The characters starting at the current cursor character and extending to the character directly above or below in the next line.

**SEN (sentence)**

User-defined entity. The default boundaries are a period, a question mark, or an exclamation point (each of which is followed by a space). You can redefine the sentence boundaries through the SET ENTITY command.

**BSEN (beginning of sentence)**

The string of characters from the cursor position to the beginning of the sentence. If you are at the beginning of a sentence, BSEN selects the entire previous sentence.

**ESEN (end of sentence)**

The string of characters from the cursor position to the end of the sentence, not including the sentence delimiter.

**PAR (paragraph)**

User-defined entity. The default boundaries are two successive line terminators. You define the paragraph boundary with the SET ENTITY command.

**BPAR (beginning of paragraph)**

The string of characters from the cursor position to the beginning of the paragraph. If you are already at the beginning of a paragraph, BPAR selects the entire previous paragraph.

**EPAR (end of paragraph)**

The string of characters from the cursor position to the end of the paragraph, including the character at the cursor, but not the paragraph delimiter.

**PAGE (page)**

User-defined entity. The default is the text between two form-feed characters, including the second form feed. You define the page boundaries with the SET ENTITY command.

**BPAGE (beginning of page)**

The text from the cursor position to the beginning of the page, not including the page delimiters.

**EPAGE (end of page)**

The text from the cursor position to the end of the page, not including the page delimiters.

**BR (beginning of range)**

The string of characters from the cursor to the beginning of the buffer.

**ER (end of range)**

The string of characters from the cursor to the end of the buffer.

**SR (select range)**

The text between the last SEL command and the cursor position. When select range is not active (no select command entered) and the cursor is at the present search string, SR selects the search string. When neither of the above conditions exists, SR selects a single character in the current direction when used with the CHGC command.

**'string'**

All characters between the initial cursor position and the beginning of 'string.'

## **EDT.4.2 Nokeypad Commands**

The following list of EDT nokeypad-editing commands presents the commands in uppercase, followed by their functions in parentheses. The word "count" preceded by "+" or "-" specifies either the number of times a nokeypad-editing command is to be repeated or the number of entities that the command will affect. (The maximum value for the count specifier is 32767.) The symbol "l" is equivalent to the word "or" in the commands that follow.

**EDT-56     EDT Editor**  
**Nokeypad Editing**

**ADV (advance)**

**ADV**

Sets the cursor's direction forward. (You can override advance for a single command by preceding that command with a minus sign.)

**APPEND (append)**

**[+ | -][count]APPEND[+ | -][count]entity[=buffer]**

Deletes the specified entities in the current buffer and moves them to the end of the specified buffer (which defaults to the PASTE buffer).

**ASC (ASCII)**

**[number]ASC**

Inserts an ASCII character when you specify the character's decimal representation (defaults to 0, or NUL).

**BACK (backup)**

**BACK**

Sets the cursor's direction backward. (You can override BACK for a single command by preceding that command with a plus sign.)

**BELL (bell)**

**BELL**

Causes the terminal bell to sound when the command is processed. (Used primarily in keypad key definitions.)

**CHGC (change case)**

**[+ | -][count]CHGC[+ | -][count]entity**

Changes the case of the characters within an entity.

**CHGL (change case lower)**

**[+ | -][count]CHGL[+ | -][count]entity**

Changes all uppercase letters within a specified entity to lowercase; existing lowercase letters remain unchanged.

**CHGU (change case upper)**

**[+ | -][count]CHGU[+ | -][count]entity**

Changes all lowercase letters within a specified entity to uppercase; existing uppercase letters remain unchanged.

**CLSS (clear search string)**

**CLSS**

Clears the search string currently in the search buffer.



**CUT (cut)**

**[+ | -][count]CUT[+ | -][count]entity[=buffer]**

Deletes the specified entities from the current buffer, moves them to the specified buffer (which defaults to the PASTE buffer), and deletes the previous contents of the receiving buffer.

**DATE (date)**

**DATE**

Inserts the current date into your text at the current cursor position.

**DELETE (delete)**

**[+ | -][count]D[+ | -][count]entity**

Deletes a specified number of entities.

**DEFK (define key)**

**DEFK**

Defines keypad operations in terms of nokeypad-editing commands.

**DESEL (deactivate select)**

**DESEL**

Cancels a select range after you have used the SEL (select) command, the SSEL (search and select) command, or the TGSEL (toggle select) command.

**DLWC (default lowercase)**

**DLWC**

Changes all uppercase letters to lowercase letters wherever you move the cursor during your EDT session. (Use the DMOV (default move) command to reset EDT so that case is not affected by move operations.)

**DMOV (default move)**

**DMOV**

Returns your editing session to EDT's default state after you have used either DLWC (default lowercase) or DUPC (default uppercase).

**DUPC (default uppercase)**

**DUPC**

Changes all lowercase letters to uppercase wherever you move the cursor during your EDT session. (Use the DMOV (default move) command to reset EDT so that case is not affected by move operations.)

**EX (exit)**

**EX**

Exits to line-editing mode.

**EDT-58     EDT Editor**  
**Nokeypad Editing**

**EXT (extend)**

**EXT**

Accepts the rest of the line (after EXT) as a line-editing command.

**FILL (fill)**

**[+ | -][count]FILL[+ | -][count]entity**

Places the maximum number of words (within the bounds set by the SET WRAP or SET SCREEN command) on a line. The default is 80 characters.

**HELP (help)**

**HELP**

Defines a different key or key sequence in keypad-editing mode to carry out the keypad HELP function.

**I (insert)**

**Itext^Z**

**I RETURN text CTRL/Z**

Inserts the text typed immediately following the I or between RETURN and CTRL/Z to the left of the cursor.

**KS (KED substitute)**

**[PASTE]KS**

Moves the cursor either to the beginning or the end of the inserted text at the completion of the nokeypad PASTE command. The placement of the cursor depends upon EDT's current direction. (There should not be a space between the command words PASTE and KS.)

**NULL (move cursor)**

**[+ | -][count]entity**

Moves the cursor to the specified entity.

**PASTE (paste)**

**[+ | -][count]PASTE[=buffer]**

Copies the contents of the specified buffer to the left of the cursor.

**QUIT (quit)**

**QUIT**

Ends the editing session without saving your edits and returns you to the system command level prompt.

**REPLACE (replace)**

**[+ | -][count]R[+ | -][count]entity**

Deletes the specified entities and replaces them with text you insert. Terminate the insertion of text with a CTRL/Z.

**REF (refresh)**

**REF**

Refreshes the entire screen.

**SEL (select)**

**SEL**

Marks the present cursor position as the beginning of a select range of text.  
Move the cursor to the desired position to mark the end of the range.

**SSEL (search and select)**

**[+ | -]SSEL[+ | -]"string"**

Both finds a string and designates it as a select range in one operation. The string must be enclosed in quotation marks.

**SHL (shift left)**

**[count]SHL**

Shifts the screen image to the left count number of tab stops, where one tab stop equals 8 columns.

**SHR (shift right)**

**[count]SHR**

Shifts the screen image to the right count number of tab stops, where one tab stop equals 8 columns.

**S (substitute)**

**[+ | -][count]S/string-1/string-2/**

Replaces one string of characters with another. The value of count defines the number of substitutions. The characters "-" (backup) and "+" (advance) specify the direction of the search. Any nonalphanumeric character will work as a delimiter (so long as it is not used in one of the strings).

**SN (substitute next)**

**[+ | -][count]SN**

Uses strings 1 and 2 defined in the last substitute command (stored in the search and substitute buffers) to replace the next occurrence of string 1 with string 2. The value of count defines the number of substitutions. The characters "-" (backup) and "+" (advance) specify the direction of the search.

**TAB (tab)**

**[count]TAB**

Moves a line of text to the next tab stop or, if a count is specified, to count number of tab stops.

**TADJ (tab adjust)**

**[+ | -][level-count]TADJ[+ | -][entity-count]entity**

Indents whole lines *level-count* number of tab stops to create layered text. A minus sign before the level-count specifier allows you to move text back toward the left margin. The entity count determines how many lines, paragraphs, or pages will be affected by the TADJ command. A minus sign before the entity or entity count means that EDT will work backward in determining which entities to indent. You must establish a SET TAB value for your EDT editing session before issuing this command.

**TC (tab compute)**

**TC**

Establishes the present cursor position and resets the indentation level count to be the quotient of the cursor position divided by the SET TAB value. The cursor position must be a multiple of the SET TAB value.

**TD (tab decrement)**

**[count]TD**

Decreases the current indentation level count by the amount indicated by the count specifier.

**TI (tab increment)**

**[count]TI**

Increases the current indentation level count by the amount indicated by the count specifier.

**TGSEL (toggle select)**

**TGSEL**

Performs the same function as the DSEL command (cancels the select range) when there is an active select range. When there is no active select range, TGSEL performs the same function as the SEL command (initiates the process of creating a select range).

**TOP (top)**

**TOP**

Places the current line at the top of the screen.

**UNDC (undelete character)**

**[count]UNDC**

Inserts the contents of the character buffer into the current buffer to the left of the cursor. The character buffer contains the last character deleted by one of the delete character commands. *Count* repeats the UNDC operation *count* number of times.

**UNDL (undelete line)**

**[count]UNDL**

Inserts the contents of the line buffer into the current text buffer just ahead of the cursor. The line buffer contains the last line deleted by one of the delete-line editing commands. *Count* repeats the UNDL operation *count* number of times.

**UNDW (undelete word)**

**[count]UNDW**

Inserts the contents of the word buffer into the current buffer to the left of the cursor. The word buffer contains the last word deleted by one of the delete word commands. *Count* repeats the UNDW operation *count* number of times.

**(circumflex)**

**[count]^[A...Z]**

Enters a control character in your text. The value of count controls the number of times the operation is performed.





# Appendix ESC

## Escape Sequences

This appendix describes the escape sequences on VT200 and VT100 series terminals; Chapter 1 explains how to use them. The escape sequences are arranged here according to function, and the following table briefly summarizes the common escape sequences.

<div>CURSOR x=integer (Default=1)</div> <div><div>up</div><div>— &lt;ESC&gt; [xA</div></div> <div><div>down</div><div>— &lt;ESC&gt; [xB</div></div> <div><div>right</div><div>— &lt;ESC&gt; [xC</div></div> <div><div>left</div><div>— &lt;ESC&gt; [xD</div></div> <div>Position</div> <div><div>row, column</div><div>— &lt;ESC&gt; [r;cH</div></div> <div><div>column, row</div><div>— &lt;ESC&gt; [c;rf</div></div>	<div>CHARACTER ATTRIBUTES</div> <div>&lt;ESC&gt; [xm where x is:</div> <div><div>0</div><div>—off</div></div> <div><div>1</div><div>—bold</div></div> <div><div>4</div><div>—underscore</div></div> <div><div>5</div><div>—blink</div></div> <div><div>7</div><div>—reverse video</div></div> <div><div>double height (top)</div><div>— &lt;ESC&gt; #3</div></div> <div><div>double height (bot)</div><div>— &lt;ESC&gt; #4</div></div> <div><div>single width (default)</div><div>— &lt;ESC&gt; #5</div></div> <div><div>double width</div><div>— &lt;ESC&gt; #6</div></div>
<div>ERASING</div> <div><div>cursor to eol</div><div>— &lt;ESC&gt; [0K</div></div> <div><div>bol to cursor</div><div>— &lt;ESC&gt; [1K</div></div> <div><div>entire line</div><div>— &lt;ESC&gt; [2K</div></div> <div><div>cursor to eos</div><div>— &lt;ESC&gt; [0J</div></div> <div><div>bos to cursor</div><div>— &lt;ESC&gt; [1J</div></div> <div><div>entire screen</div><div>— &lt;ESC&gt; [2J</div></div>	<div>CHARACTER SETS</div> <div><div>on</div><div>off</div></div> <div><div>US</div><div>&lt;ESC&gt; (B</div><div>&lt;ESC&gt; )B</div></div> <div><div>UK</div><div>&lt;ESC&gt; (A</div><div>&lt;ESC&gt; )A</div></div> <div><div>GRAPHIC</div><div>&lt;ESC&gt; (0</div><div>&lt;ESC&gt; )0</div></div>
<div>SCROLLING REGION</div> <div>&lt;ESC&gt; [t;br (t &lt; b)</div>	<div>RESET</div> <div>&lt;ESC&gt; c</div>

ZK-1763-84

**NOTE:** If your terminal is not in the VT200 series or VT100 series, consult your terminal user guide for the appropriate escape sequences for your terminal.

## ESC-2    **Escape Sequences**

### **Moving the Cursor**

#### **ESC.1    Moving the Cursor**

The following table lists the escape sequences that allow you to move the cursor, where  $n$  represents an integer that determines the number of character positions or lines that the cursor is to move. If you omit  $n$ , the number defaults to one. In the escape sequence that positions the cursor,  $r$  and  $c$  represent integers that indicate the row number and the column number, respectively.

Escape Sequence	Function
<ESC> [nA	Move cursor up; no scrolling
<ESC> M	Move cursor up; scrolls within scrolling region
<ESC> [nB	Move cursor down; no scrolling
<ESC> D <LF>	Move cursor down; scrolls within scrolling region
<ESC> [nC	Move cursor right
<ESC> [nD	Move cursor left
<ESC> E	Move cursor to the beginning of the next line
<ESC> [r;cH	Position cursor on the screen
<ESC> 7	Save cursor's column position and character attributes
<ESC> 8	Restore cursor's column position and character attributes

A carriage return moves the cursor to the beginning of the next line. Therefore, if you want text to appear on the screen at a certain place, be sure that the text immediately follows the escape sequence that positions the cursor. Putting a carriage return between the escape sequence and the text causes the text to be written at the beginning of the next line.

#### **ESC.2    Setting Character Sets and Characteristics**

Escape sequences allow you to choose a character set from a given group of character sets. You can choose character sets permanently (using locking shifts) or for the insertion of just the next character you enter (using single shifts).

## **ESC.2.1 Specifying Character Sets**

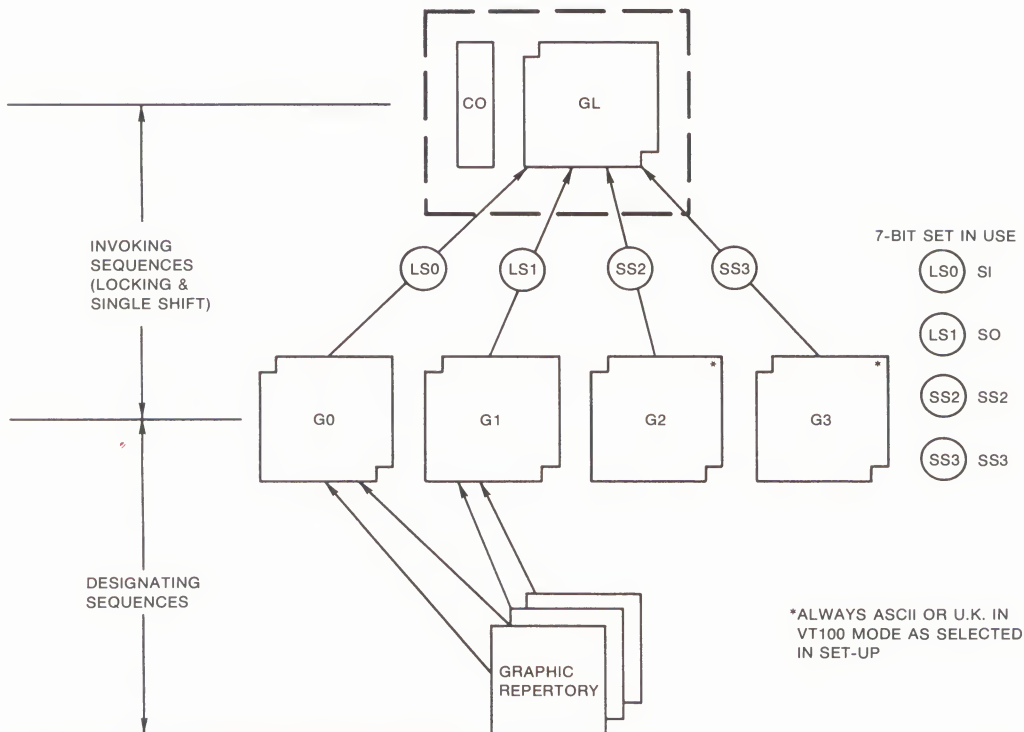
The VT200 and VT100 series terminals allow you to specify three different character sets:

1. US (default)—The ASCII character set. The escape sequence `<ESC> (B` specifies the US character set.
2. UK—The same as the US character set with the exception that pressing SHIFT/3 echoes the pound sign used in the United Kingdom rather than the number sign used in the United States. The escape sequence `<ESC> (A` specifies the UK character set.
3. Graphic—The graphic character set echoes a different set of characters for ASCII codes 5F through 7E. The escape sequence `<ESC> (0` specifies the graphic character set.

On VT200 series terminals it is also possible to specify downline loadable (or “soft”) character sets. For information on designating soft character sets, refer to your terminal’s reference manual.

Selecting a character set is a two-step process. The following figure illustrates the procedure for selecting character sets.

## ESC-4    Escape Sequences Setting Character Sets and Characteristics



7K 1773 84

To select a character set from the graphic repertory, you must first use character set selection sequences to designate a graphic set. This marks the graphic set and makes it available (or "on call") for use later in your program. You may designate up to two character sets each for both locking shift and single shift operations; locking shifts designate G0 and G1 and single shifts designate G2 and G3. Character sets remain designated until you enter another character set selection escape sequence. The following table lists the escape sequences you use to designate character sets.

## Escape Sequences    ESC-5

### Setting Character Sets and Characteristics

Character Set	Escape Sequence	Designate as:
ASCII	ESC(B	G0
	ESC)B	G1
U.K. National	ESC(A	G0
	ESC)A	G1
Graphic	ESC(0	G0
	ESC)0	G1

To actually map any one of these sets into GL, you must then invoke any of G0 through G3 into GL. The GL block contains the graphic characters with decimal values 33 through 127; these are the only characters that are affected when you invoke different character sets. The CO (Control) block contains graphic characters with decimal values 0 through 32; these characters are not affected by invoking different character sets. You invoke character sets using either locking shifts or single shifts, depending upon whether you want to select a particular character set for extended use or for the insertion of just a single character. When using locking shifts, you should always invoke G0 first, leaving G1 available for the designation of another character set. The following table lists the locking shift and single shift control functions used to invoke character sets.

Control Name	Coding	Function
LS0 - Lock Shift G0	SI	Invoke G0 into GL (default)
LS1 - Lock Shift G1	SO	Invoke G1 into GL
SS2 - Single Shift G2	SS2 (or ESC N)	Invoke G2 into GL
SS3 - Single Shift G3	SS3 (or ESC O)	Invoke G3 into GL

All locking shifts remain active until you issue another locking shift; single shifts remain active for only the next single graphic character. The default graphic character set mapping is reset whenever you turn on your terminal.

## ESC-6    **Escape Sequences**

### **Setting Character Sets and Characteristics**

#### **ESC.2.2 Specifying Display Characteristics**

The following table lists the escape sequences that allow you to specify the display characteristics. (When programming, you should use the screen package rather than escape sequences.)

Escape Sequence	Function
<ESC> [0m (default)	Write normal characters
<ESC> [1m	Write bold characters
<ESC> [4m	Write underlined characters
<ESC> [5m	Write blinking characters
<ESC> [7m	Write reverse video characters
<ESC> #3	Write double height (top half)
<ESC> #4	Write double height (bottom half)
<ESC> #5	Write single width (default)
<ESC> #6	Write double width
<ESC> 7	Save cursor's column position and character attributes
<ESC> 8	Restore cursor's column position and character attributes

**NOTE:** The height functions cannot be used with the graphic character set.

#### **ESC.3 Erasing the Screen**

The following table lists the escape sequences that allow you to erase specified sections of text. After text is erased, the cursor is left where the erasing completed (for example, erasing from cursor to end of line leaves the cursor at the end of the line, erasing from beginning of line to cursor leaves the cursor at its original position). Erasing text does not affect the current character attributes.

Escape Sequence	Erase Function
<ESC> [0K	From cursor to end of line
<ESC> [1K	From beginning of line to cursor
<ESC> [2K	Entire line
<ESC> [0J	From cursor to end of screen
<ESC> [1J	From bottom of screen to cursor
<ESC> [2J	Entire screen



## ESC.4    Creating a Scrolling Region

A scrolling region is a horizontal section of the screen within which the text scrolls. The text above and below the scrolling region remains stationary. The following table lists the escape sequences that allow you to create a scrolling region and to define whether cursor position 0,0 means the upper left corner of the screen or the upper left corner of the scrolling region.

In the first escape sequence, t and b represent integers that indicate the first and last lines of the scrolling region. Both integers must be between 1 and 24; t must be less than b.

Escape Sequence	Function
<ESC> [t;br	Set up a scrolling region
<ESC> [?6h	Set cursor position 0,0 equal to the upper left corner of the scrolling region
<ESC> [?6l (default)	Set cursor position 0,0 equal to the upper left corner of the screen

## ESC.5    Setting Terminal Characteristics

Section 1.3 describes the setup functions that allow you to set the terminal attributes. Many of these attributes can also be set with escape sequences.

The RESET key used in setup to reset the default terminal attributes can be duplicated with the <ESC> escape sequence.

The following table lists the escape sequences that allow you to set the terminal attributes that you would normally set from the Display and Tab Set-Up screens on VT200 series terminals or from SET-UP A on VT100 series terminals.

Escape Sequence	Function
<ESC> [?3l	Set screen width to 80 columns
<ESC> [?3h	Set screen width to 132 columns
<ESC> H	Set tab at current column
<ESC> [g or <ESC> [0g	Clear tab at current column
<ESC> [3g	Clear all tabs

## ESC-8    **Escape Sequences**

### **Setting Keypad Characteristics**

## **ESC.6    Setting Keypad Characteristics**

On both the VT200 and VT100 series terminals, the auxiliary keypad to the right of the main keyboard and the four arrow keys to the left of that keypad can be set to generate data or escape sequences. By default, the arrow keys and the PF<sub>n</sub> keys (on the top row of the keypad) generate escape sequences, and the remaining keypad keys generate data.

To make the keypad generate numeric values, use the SET TERMINAL/NUMERIC command or the <ESC> > escape sequence. To make the keypad keys generate application values, use the SET TERMINAL/APPLICATION command or the <ESC> = escape sequence.

The following table lists the code generated by the auxiliary keypad keys for both the numeric and application keys. The values generated by these keys are identical on VT200 and VT100 series terminals.

Key	Numeric	Application
0	0	<ESC> Op
1	1	<ESC> Oq
2	2	<ESC> Or
3	3	<ESC> Os
4	4	<ESC> Ot
5	5	<ESC> Ou
6	6	<ESC> Ov
7	7	<ESC> Ow
8	8	<ESC> Ox
9	9	<ESC> Oy
-	- (minus)	<ESC> Om
,	, (comma)	<ESC> Ol
.	. (period)	<ESC> On
ENTER	RETURN	<ESC> OM
PF1	<ESC> OP	<ESC> OP
PF2	<ESC> OQ	<ESC> OQ
PF3	<ESC> OR	<ESC> OR
PF4	<ESC> OS	<ESC> OS

The following table lists the codes generated by the arrow keys in both Cursor Key Mode Set and Cursor Key Mode Reset.

## Escape Sequences    ESC-9

### Setting Keypad Characteristics

Key	Cursor Key Mode Reset	Cursor Key Mode Set
Up arrow	<ESC> [A	<ESC> OA
Down arrow	<ESC> [B	<ESC> OB
Left arrow	<ESC> [C	<ESC> OC
Right arrow	<ESC> [D	<ESC> OD

In addition to the keys described above, the VT200 series terminal has a set of six editing keys located between the main keyboard and the auxiliary keypad and a row of 20 function keys located above the main keyboard. The first five function keys (F1 through F5) do not transmit codes; they are local keys. Function keys F6 through F14 are VMS-defined. (See Appendix KEY for a table listing these operating system-defined keys.) Applications that wish to make use of keys F6 through F14 (to generate the code listed in the table below) should insert the following command into their \$QIO function code:

**IO\$\_READVBLK ! IO\$\_M\_NOFILTR**

By default, function keys F15 (Help), F16 (Do), and F17 through F20 generate the codes listed below.

The following table lists the editing keypad keys and function keys along with the codes they generate.

Generic Key Name	Code Generated
Find (E1)	<ESC> [1~
Insert Here (E2)	<ESC> [2~
Remove (E3)	<ESC> [3~
Select (E4)	<ESC> [4~
Prev Screen (E5)	<ESC> [5~
Next Screen (E6)	<ESC> [6~
F6	<ESC> [17~
F7	<ESC> [18~
F8	<ESC> [19~
F9	<ESC> [20~
F10	<ESC> [21~
F11	<ESC> [23~
F12	<ESC> [24~
F13	<ESC> [25~
F14	<ESC> [26~

## ESC-10    **Escape Sequences**

### **Setting Keypad Characteristics**

Generic Key Name	Code Generated
Help (F15)	<ESC> [28~
Do (F16)	<ESC> [29~
F17	<ESC> [31~
F18	<ESC> [32~
F19	<ESC> [33~
F20	<ESC> [34~

# Appendix EXP

## Expressions

The following table lists data operations and comparisons in order of precedence, beginning with the highest.

Operator	Precedence	Description
+	1	Indicates a positive number
-	1	Indicates a negative number
*	2	Multiplies two numbers
/	2	Divides two numbers
+	3	(1) Adds two numbers      (2) Concatenates two character strings
-	3	(1) Subtracts two numbers (2) Subtracts two character strings
.EQS.	4	Tests if two character strings are equal
.GES.	4	Tests if first character string is greater than or equal
.GTS.	4	Tests if first character string is greater than
.LES.	4	Tests if first character string is less than or equal
.LTS.	4	Tests if first character string is less than
.NES.	4	Tests if two character strings are not equal
.EQ.	4	Tests if two numbers are equal
.GE.	4	Tests if first number is greater than or equal to
.GT.	4	Tests if first number is greater than
.LE.	4	Tests if first number is less than or equal to
.LT.	4	Tests if first number is less than
.NE.	4	Tests if two numbers are not equal
.NOT.	5	Logically negates a number
.AND.	6	Combines two numbers with a logical AND
.OR.	7	Combines two numbers with a logical OR

The following tables demonstrate the results of logical operations on a bit-by-bit basis and a number-by-number basis. In logical operations, a character string beginning

## EXP-2 Expressions

with an uppercase or lowercase T or Y is treated as the number 1; a character string beginning with any other character is treated as the number 0. In logical operations, odd numbers are considered true and even numbers and zero are considered false.

Given Bit A	Bit B	Results .NOT A	A .AND. B	A .OR. B
1	1	0	1	1
1	0	0	0	1
0	1	1	0	1
0	0	1	0	0

Given Number A	Number B	Results .NOT A	A .AND. B	A .OR. B
odd	odd	even	odd	odd
odd	even	even	even	odd
even	odd	odd	even	odd
even	even	odd	even	even



# Appendix KEY

## Terminal Keys

The following tables present the operating system's interpretation of keys on terminals in the VT200 and VT100 series.

### KEY.1 VT200 Terminal Series

The following table describes how the operating system responds when various keys and control characters are pressed on VT200 series terminals. The table assumes that line editing is enabled (the default). (Characters not mentioned in the table are treated as null characters.)

Character	HEX	System Response
CTRL/A	01	Switches between overstrike and insert modes
CTRL/B	02	Recalls previous line
CTRL/C	03	Interrupts current image (image may define alternate CTRL/C action)
CTRL/D	04	Moves cursor left one character
CTRL/E	05	Moves cursor to end of line
CTRL/F	06	Moves cursor right one character
CTRL/H	08	Moves cursor to beginning of line
CTRL/I	09	Horizontal tab
CTRL/J	0A	Deletes previous word
CTRL/M	0D	Line terminator
CTRL/O	0F	Suspends/resumes echoing of output
CTRL/Q	11	Resumes output (see CTRL/S)
CTRL/R	12	Refreshes current line
CTRL/S	13	Suspends output (see CTRL/Q)

## KEY-2 Terminal Keys

### VT200 Terminal Series

Character	HEX	System Response
CTRL/T	14	Displays process information (must be enabled with SET CONTROL=T command)
CTRL/U	15	Deletes characters from cursor to beginning of line
CTRL/V	16	Passes next character or escape sequence to the image without interpreting it as described in this table
CTRL/X	18	Purges type-ahead buffer; if characters are on the current line, deletes characters from cursor to beginning of line
CTRL/Y	19	Interrupts current image
CTRL/Z	1A	Indicates end of file
Data keys	-	Enter appropriate character
⌫	-	Deletes previous character
CTRL	-	Modifies another key
CTRL/[ (ESC)	1B	Begins escape sequence
CTRL/F5	-	Executes answerback message
DOWN ARROW	-	Repeats current line
F1 (No Scroll)	-	Suspends/resumes output
F5 (Break)	-	Shuts down transmission line
F6 (Interrupt)	-	Interrupts the current image
F10 (Exit)	-	Terminates the current image or command procedure
F12 (Backspace)	08	Moves cursor to beginning of line
F13 (Line Feed)	-	Deletes previous word
F14 (^A)	01	Switches between overstrike and insert modes
LEFT ARROW	-	Moves cursor left one character
PFn	-	Can be defined (see DEFINE/KEY)
RETURN	-	Line terminator
RIGHT ARROW	-	Moves cursor right one character
TAB	-	Horizontal tab
UP ARROW	-	Repeats current line

## KEY.2 VT100 Terminal Series

The following table describes how the operating system responds when various keys and control characters are pressed on VT100 terminals. The table assumes that line editing is enabled (the default). (Characters not mentioned in the table are treated as null characters.)

# Terminal Keys      KEY-3

## VT100 Terminal Series

Character	HEX	System Response
CTRL/A	01	Switches between overstrike and insert modes
CTRL/B	02	Recalls previous line
CTRL/C	03	Interrupts current image (image may define alternate CTRL/C action)
CTRL/D	04	Moves cursor left one character
CTRL/E	05	Moves cursor to end of line
CTRL/F	06	Moves cursor right one character
CTRL/H	08	Moves cursor to beginning of line
CTRL/I	09	Horizontal tab
CTRL/J	0A	Deletes previous word
CTRL/M	0D	Line terminator
CTRL/O	0F	Suspends/resumes echoing of output
CTRL/Q	11	Resumes output (see CTRL/S)
CTRL/R	12	Refreshes current line
CTRL/S	13	Suspends output (see CTRL/Q)
CTRL/T	14	Displays process information
CTRL/U	15	Deletes characters from cursor to beginning of line
CTRL/V	16	Passes next character or escape sequence to the image without interpreting it as described in this table
CTRL/X	18	Purges type-ahead buffer; if characters are on the current line, deletes characters from cursor to beginning of line
CTRL/Y	19	Interrupts current image
CTRL/Z	1A	Indicates end of file
Data keys	-	Enter appropriate character
Backspace (^H)	08	Moves cursor to beginning of line
BREAK	-	Shuts down transmission line
CTRL	-	Modifies another key
CTRL/BREAK	-	Executes answerback message
DELETE	-	Deletes previous character
DOWN ARROW	-	Repeats current line
ESC	1B	Begins escape sequence
LEFT ARROW	-	Moves cursor left one character
LINE FEED	-	Deletes previous word
NO SCROLL	-	Suspends/resumes output
PFn	-	Can be defined (see DEFINE/KEY)

**KEY-4    Terminal Keys**  
**VT100 Terminal Series**

Character	HEX	System Response
RETURN	-	Line terminator
RIGHT ARROW	-	Moves cursor right one character
TAB	-	Horizontal tab
UP ARROW	-	Repeats current line

## Appendix LEX

### Lexical Functions

The following tables list the lexical functions by category. The remainder of the appendix describes each function in alphabetical order.

Integer/character string conversion:

Function	Description
F\$TYPE	Determines whether a symbol is a number or a string
F\$INTEGER	Converts a character string to a number
F\$STRING	Converts a number to a character string
F\$CVSI	Converts a bit string to a signed integer
F\$CVUI	Converts a bit string to an unsigned integer

Character string manipulation:

Function	Description
F\$LENGTH	Returns the length of a character string
F\$LOCATE	Returns the location of a substring
F\$EXTRACT	Returns a substring
F\$ELEMENT	Returns one of a list of elements

Text formatting:

Function	Description
F\$EDIT	Edits a string
F\$FAO	Creates character strings from character strings, numbers, and special directives

## LEX-2 Lexical Functions

Time:

Function	Description
F\$TIME	Returns the current date and time
F\$CVTIME	Converts a date/time for comparison

Devices and files:

Function	Description
F\$DIRECTORY	Returns the name of the default directory
F\$FILE_ATTRIBUTES	Returns information on a file
F\$GETDVI	Returns information on a device
F\$IDENTIFIER	Returns the numeric equivalent of an alphanumeric identifier or vice versa
F\$PARSE	Returns a file specification given a partial specification and defaults
F\$SEARCH	Returns specific file specifications given a file specification containing wildcards
F\$TRNLNM	Returns the equivalence string or attributes of a logical name

Environment:

Function	Description
F\$ENVIRONMENT	Returns information about the DCL command environment
F\$GETJPI	Returns information about a process
F\$PROCESS	Returns the name of the current process
F\$PRIVILEGES	Returns the current process privileges
F\$SETPRV	Sets process privileges
F\$USER	Returns the UIC of the current process in alphanumeric format
F\$PID	Returns process identification numbers of processes using the system
F\$MODE	Returns the process type—interactive, batch, or network
F\$VERIFY	Returns or sets the verification mode
F\$MESSAGE	Returns a system message given the code
F\$GETSYI	Returns information about the system



**F\$CVSI (start-bit,number-of-bits,string)**

Converts the specified bits in the specified character string to a signed number.

**ARGUMENTS****start-bit**

The offset of the starting bit.

**number-of-bits**

The length of the bit string, which must be less than or equal to the number of bits in the string.

**string**

The character string to be edited.

The following example converts the low-order four bits of the ASCII character + (the bit configuration of this character is 00101011—hexadecimal 2B) to a signed number.

```
$ LOW_FOUR = F$CVSI (0,4,"+")
$ SHOW SYMBOL LOW_FOUR
LOW_FOUR = -5    Hex = FFFFFFFB    Octal = 177773
```

**F\$CVTIME ([input-time],[format],[field])**

Given a character string containing a time, F\$CVTIME returns the time or a field of the time in the specified format. If you omit arguments, commas to the left of the last specified argument must be included as place holders.

**ARGUMENTS****input-time**

A string containing an absolute, combination, or delta time, or TODAY, TOMORROW, or YESTERDAY. (Section 5.4 describes absolute, delta, and combination formats.) If *input-time* is omitted or specified as a null string (""), the current system date and time, in absolute format, is used. If parts of the date field are omitted, the missing values default to the current date. If parts of the time field are omitted, the missing values default to zero.

**format**

A character string containing one of the following (do not abbreviate): ABSOLUTE, COMPARISON (default), or DELTA. Comparison format ("yyyy-mm-dd hh:mm:ss.cc") is used for comparing two times. If *input-time* is a delta time, you must specify DELTA. If *input-time* is an absolute or combination time, *format* can be either ABSOLUTE or COMPARISON.

## LEX-4 Lexical Functions

### F\$CVTIME

#### field

A character string containing one of the following (do not abbreviate): DATE, MONTH, DATETIME (default), SECOND, DAY, TIME, HOUR, WEEKDAY, HUNDREDTH, YEAR, MINUTE. If *input-time* is a delta time and *format* is DELTA, you cannot specify MONTH, WEEKDAY, or YEAR.

The following example returns the current system time in comparison format.

```
$ TIME = F$CVTIME ()
$ SHOW SYMBOL TIME
TIME = "1984-10-15 20:42:03.21"
```

The next example shows which weekday May 6, 1984 lands on:

```
$ DAY = F$CVTIME ("6-MAY-1984", , "WEEKDAY")
$ SHOW SYMBOL DAY
DAY = "Sunday"
```

### F\$CVUI (start-bit,number-of-bits,string)

Converts the specified bits in the specified character string to an unsigned number.

#### ARGUMENTS

##### start-bit

The offset of the starting bit.

##### number-of-bits

The length of the bit-string, which must be less than or equal to the number of bits in *string*.

##### string

The character string to be edited.

The following example converts the low-order four bits of the ASCII character + (the bit configuration of this character is 00101011—hexadecimal 2B) to an unsigned number.

```
$ LOW_FOUR = F$CVUI (0,4,"+")
$ SHOW SYMBOL LOW_FOUR
LOW_FOUR = 11    Hex = 0000000B    Octal = 000013
```

## F\$DIRECTORY ( )

Returns the current default directory as a character string, as shown in the following example.

```
$ DEFAULT = F$DIRECTORY ( )
$ SHOW SYMBOL DEFAULT
   DEFAULT = "[ACCOUNTS]"
```

## F\$EDIT (string,edit-list)

Edits the character string as specified by edit-list.

### ARGUMENTS

#### string

A character string to be edited. Quoted sections of the string are not edited.

#### edit-list

A character string containing one or more of the following keywords (do not abbreviate, separate keywords with commas).

Edit	Action
COLLAPSE	Removes all spaces or tabs
COMPRESS	Replaces multiple spaces or tabs with a single space
LOWERCASE	Changes all uppercase characters to lowercase
TRIM	Removes leading and trailing spaces or tabs
UNCOMMENT	Removes comments
UPCASE	Changes all lowercase characters to uppercase

The following example compresses the line and removes leading and trailing spaces.

```
$ LINE = "  Could it have been a clearer day?  "
$ LINE = F$EDIT (LINE, "COMPRESS,TRIM")
$ SHOW SYMBOL LINE
   LINE = "Could it have been a clearer day?"
```

## F\$ELEMENT (number,delimiter,string)

Extracts one element from a string of elements.

## LEX-6 Lexical Functions

### F\$ELEMENT

#### ARGUMENTS

##### number

The number of the element to extract (numbering begins with zero). If *number* exceeds the number of elements in the string, F\$ELEMENT returns the delimiter.

##### delimiter

A character used to separate the elements.

##### string

A string containing the delimited list of elements.

The following command procedure processes files named CHAP1, CHAP2, ... CHAP6, CHAPA, CHAPB, and CHAPC, in that order. (0 is included in the CHAPTERS string to clarify the procedure logic).

```
$ ! INDEX.COM
$ !
$ CHAPTERS = "0,1,2,3,4,5,6,A,B,C"
$ NEXT = 0
$ LOOP:
$   NEXT = NEXT + 1
$   NUM = F$ELEMENT(NEXT, ",", CHAPTERS)
$   RUN INDEX CHAP'NUM'
$   IF (CHAPTERS .NES. ",") THEN GOTO LOOP
```

### F\$ENVIRONMENT (item)

Returns information about the current DCL command environment.

#### ARGUMENTS

##### item

A character string containing one of the following (do not abbreviate):

Item	Data	Information Returned
CAPTIVE	string	TRUE if you are logged into a captive account.
CONTROL	string	Control characters currently enabled with SET CONTROL. Multiple characters are separated by commas; if no control characters are enabled, the null string is returned.
DEFAULT	string	Current default device and directory.
DEPTH	integer	Current command procedure depth.

# Lexical Functions    LEX-7

## F\$ENVIRONMENT

Item	Data	Information Returned
INTERACTIVE	string	TRUE if the process is executing interactively.
KEY_STATE	string	Current locked keypad state; see the DEFINE /KEY command in Appendix DCL.
MAX_DEPTH	integer	Maximum allowable command procedure depth.
MESSAGE	string	Current setting of SET MESSAGE qualifiers; each qualifier is prefaced by a slash.
NOCONTROL	string	Currently disabled control characters. Multiple characters are separated by commas; if no control characters are disabled, the null string is returned.
ON_CONTROL_Y	string	TRUE if ON_CONTROL_Y is set. ON_CONTROL_Y always returns FALSE at DCL command level.
ON_SEVERITY	string	Severity level at which the action specified with the ON command is performed. ON_SEVERITY returns NONE when SET NOON is in effect or at DCL command level.
OUTPUT_RATE	string	Delta time indicating how often data is written to the batch job log file. OUTPUT_RATE returns a null string if used interactively.
PROCEDURE	string	File specification of the current command procedure. PROCEDURE returns a null string if used interactively.
PROMPT	string	Current DCL prompt.
PROMPT_CONTROL	string	TRUE if a carriage return and line feed precede the prompt.
PROTECTION	string	Current default file protection.
SYMBOL_SCOPE	string	[NO]LOCAL,[NO]GLOBAL to indicate the current symbol scoping state.
VERIFY_IMAGE	string	TRUE if image verification is in effect; see SET VERIFY=IMAGE in Appendix DCL.
VERIFY_PROCEDURE	string	TRUE if procedure verification is in effect; see SET VERIFY=PROCEDURE in Appendix DCL.

## LEX-8    **Lexical Functions**

### **F\$ENVIRONMENT**

The following command procedure saves a user's default device, directory, and protection, executes a series of commands that may change those defaults, and then restores the original defaults.

```
$ ! PROCEDURE.COM
$ !
$ OLD_DEFAULT = F$ENVIRONMENT("DEFAULT")
$ OLD_PROT = F$ENVIRONMENT("PROTECTION")
.
. ! commands
.
$ SET DEFAULT 'OLD_DEFAULT'
$ SET PROTECTION=('OLD_PROT')/DEFAULT
```

### **F\$EXTRACT (start,length,string)**

Extracts the specified characters from the specified string.

#### **ARGUMENTS**

##### **start**

The offset of the starting character.

##### **length**

The number of characters to extract; must be less than or equal to the size of the string.

##### **string**

The character string to be edited.

The following example (which would be in a command procedure) reads a character string from the terminal and extracts the first character.

```
$ ! PROCEDURE.COM
$ !
$ INQUIRE YN /NOPUNCTUATION "Do you want to continue? "
$ YN = F$EXTRACT (0,1,YN)
.
.
.
```

### **F\$FAO (format-instructions,[data-entity,...])**

Creates character strings from character and numeric input (FAO stands for formatted ASCII output). Formatting instructions convert numbers to character strings, insert carriage returns and form feeds, insert text, and so on.



## ARGUMENTS

### **format-instruction**

Character string (enclose in quotation marks or equate to a symbol) consisting of text and directives. Directives take the following forms:

!code	One directive
!repeat(code)	A directive repeated a specified number of times
!length-code	A directive affecting an output field of a specified length
!repeat(length-code)	A directive affecting an output field of a specified length, repeated a specified number of times

### **data-entity**

Arguments required by the FAO directives used in the control string. Specify the data entities as integer or character string expressions.

The directives and their parameter requirements are as follows:

Instruction	Description	Parameter
<b>Character string insertion</b>		
!AS	Inserts a character string as is. The field length defaults to the length of the character string. Short values inserted in explicit-length fields are left-justified and blank filled. Long values inserted in explicit-length fields are truncated on the right	Character string
<b>Zero-filled numeric conversion</b>		
!OB !OW !OL	Converts a byte, word, or longword to octal notation <sup>1</sup>	A number. For byte and word conversions only the low-order 8 and 16 bits are used.
!XB !XW !XL	Converts a byte, word or longword to hexadecimal notation. <sup>1</sup>	

<sup>1</sup> The output field lengths default to 2 (byte), 4 (word), and 8 (longword) for hexadecimal; 3, 6, and 11 for octal; and the required number of characters for decimal.

The numbers in the output field are right-justified and zero-filled on the left. For hexadecimal and octal numbers: explicit-length fields longer than the default are blank-filled on the left; explicit-length fields shorter than the default are truncated on the left. For decimal numbers: explicit-length fields longer than the default are zero-filled on the left; explicit-length fields shorter than the default are filled with asterisks.

# LEX-10    Lexical Functions

## F\$FAO

### Zero-filled numeric conversion

!ZB	Converts a byte, word, or longword to decimal notation. <sup>1</sup>
!ZW	
!ZL	

### Blank-filled numeric conversion

!UB	Converts a byte, word, or longword to decimal notation without adjusting for negative numbers. <sup>2</sup>	A number. For byte and word conversions, only the low-order 8 and 16 bits are used.
!UW		
!UL		
!SB	Converts a byte, word, or longword to decimal notation with negative numbers converted properly. <sup>2</sup>	
!SW		
!SL		

<sup>1</sup>The output field lengths default to 2 (byte), 4 (word), and 8 (longword) for hexadecimal; 3, 6, and 11 for octal; and the required number of characters for decimal.

The numbers in the output field are right-justified and zero-filled on the left. For hexadecimal and octal numbers: explicit-length fields longer than the default are blank-filled on the left; explicit-length fields shorter than the default are truncated on the left. For decimal numbers: explicit-length fields longer than the default are zero-filled on the left; explicit-length fields shorter than the default are filled with asterisks.

<sup>2</sup>Output field lengths default to the required number of characters. Values shorter than explicit-length fields are right-justified and blank-filled. Values longer than explicit-length fields cause the field to be filled with asterisks.

Instruction	Description	Parameter
<b>Special formatting</b>		
!/\	Inserts a carriage return.	None.
!_	Inserts a tab.	
!^	Inserts a form feed.	
!!	Inserts an exclamation point.	
!%I	Converts a longword integer to an alphanumeric UIC in the format [group,member].	Integer
!%S	Inserts an uppercase S if the most recently converted number is not 1.	
!%U	Converts a longword integer to a numeric UIC in the format [group,member]. The directive inserts the brackets and the comma.	Integer
!n <...!>	Left justifies and blank fills all data represented by the ellipsis in fields n characters wide.	
!n*c	Repeats the character represented by c n times.	

<b>Time</b>		
!%T	Inserts the current time.	A literal 0.
!%D	Inserts the current date/time.	A literal 0.
<b>Parameter Interpretation</b>		
!-	Reuses the last parameter.	None.
!+	Skips the next parameter.	

For example, !SL means convert a number to a character string; !2(SL) means convert two numbers; !8SL means convert a number and put it in an 8-character field; and !2(8SL) means convert two numbers and put them in 8-character fields. Character data other than directives can be mixed with the directives and appears in the output as it is typed. For example, COMPLAINTS!8SL creates a character string consisting of the word COMPLAINTS followed by a converted number in an 8-character field. To create a literal exclamation point, type a double exclamation point.

You can specify the repeat and length portions of a directive as literal numbers or in variable format by substituting a number sign. The number sign indicates that the repeat or length number is the value of the next parameter.

For each directive that works upon a data entity, you must supply the data entity as a literal or symbol in the parameter list. The following command, for example, converts the number equated to COMPLAINT\_COUNT to a character string.

```
$ REPORT_1 = F$FAO("!SL",COMPLAINT_COUNT)
```

Parameters must correspond exactly to the order of their respective directives. Remember that repeat counts require the specified number of parameters. In the following example, the literal "Complaints" corresponds to the formatting instruction !AS, while COMPLAINT\_COUNT\_1 and COMPLAINT\_COUNT\_2 correspond to the instruction !2(8SL).

```
$ REPORT_1 = F$FAO("!AS!2(8SL)",)-  
_ $ "Complaints",COMPLAINT_COUNT_1,COMPLAINT_COUNT_2
```

If you misplace a parameter, you will probably not receive an error message, but the output will be incorrect.

## **F\$FILE\_ATTRIBUTES (file-spec,item)**

Returns a specified item of information about a specified file.

## LEX-12 Lexical Functions

### F\$FILE\_ATTRIBUTES

#### ARGUMENTS

##### **file-spec**

A character string containing a file specification with no wildcards.

##### **item**

A character string containing one of the following (do not abbreviate):

Item	Data Type	Information Returned
ALQ	integer	Allocation quantity
BDT	string	Backup date/time
BKS	integer	Bucket size
BLS	string	Block size
CBT	string	TRUE if contiguous-best-try
CDT	string	Creation date/time
CTG	string	TRUE if contiguous
DEQ	integer	Default extension quantity
DID	string	Directory identification
DVI	string	Device name
EDT	string	Expiration date/time
EOF	integer	Number of blocks used
FID	string	File identification
FSZ	integer	Fixed control area size
GRP	integer	Owner group number
KNOWN	string	TRUE if file is installed.
MBM	integer	Owner member number
MRN	integer	Maximum record number
MRS	integer	Maximum record size
NOA	integer	Number of areas
NOK	integer	Number of keys
ORG	string	File organization (SEQ, REL, IDX)
PRO	string	File protection
PVN	integer	Prologue version number

Item	Data Type	Information Returned
RAT	string	Record attributes (CR, PRN, FTN)
RCK	string	TRUE if read check
RDT	string	Revision date/time
RFM	string	Record format (VAR, FIX, VFC, UDF)
RVN	integer	Revision number
UIC	string	Owner UIC
WCK	string	TRUE if write check

The following example returns the file identification of 1983.DAT.

```
$ FILEID = F$FILE_ATTRIBUTES ("1983.DAT", "FID")
$ SHOW SYMBOL FILEID
FILEID = "(65,1,0)"
```

### F\$GETDVI (device-name,item)

Returns a specified item of information about a specified device.

#### ARGUMENTS

##### device-name

A character string containing a physical device name or a logical name equated to a physical device name.

##### item

A character string containing one of the following (do not abbreviate):

Item	Data Type	Information Returned
ACPPID	string	ACP process identification
ACPTYPE	integer	ACP type code
ALL	string	TRUE if device is allocated
ALLDEVNAM	string	Allocation class device name
AVL	string	TRUE if device is available
CCL	string	TRUE if carriage control device
CLUSTER	integer	Volume cluster size
CONCEALED	string	TRUE if device is a concealed device
CYLINDERS	integer	Number of cylinders on the volume
DEVBUFSIZ	integer	Device buffer size

**LEX-14      Lexical Functions**  
**F\$GETDVI**

Item	Data Type	Information Returned
DEVCHAR	integer	Device characteristics
DEVCHAR2	integer	More device characteristics
DEVCLASS	integer	Device class (value): Disk device (1), Tape (2), Synchronous communications device (32), Terminal (66), Real-time (96), Bus (128), Mailbox (160), Miscellaneous device (200)
DEVDEPEND	integer	Device dependent information
DEVDEPEND2	integer	More device dependent information
DEVLOCKNAM	string	Device lock name
DEVNAM	string	Device name
DEVSTS	integer	Device dependent status information
DEVTYPE	integer	Device type (value): KLESI (5), TK50P (8), RQDX1 (7), RQDX2 (7), RL02 (10), TK50 (10), RRD50 (13), DEQNA (22), DMV11 (23), Removable RC25 (23), Fixed RC25 (24), RD51 (25), RX50 (26), RD52 (27), RD53 (28), CDR50 (34), VT100 (96), VT240 (110)
DIR	string	TRUE if device is directory structured
DMT	string	TRUE if device is marked for dismount
DUA	string	TRUE if the device is a generic device
ELG	string	TRUE if error logging is enabled
ERRCNT	integer	Error count
EXISTS	string	TRUE if the device exists on the system
FOD	string	TRUE if file-oriented device
FOR	string	TRUE if device is mounted foreign
FREEBLOCKS	integer	Free blocks left on the volume
FULLDEVNAM	string	Fully qualified device name
GEN	string	TRUE if device is generic
IDV	string	TRUE if device is capable of input
LOGVOLNAM	string	Logical volume name
MAXBLOCK	integer	Number of logical blocks on the volume
MAXFILES	integer	Maximum files on volume
MBX	string	TRUE if device is a mailbox
MNT	string	TRUE if device is mounted
MOUNTCNT	integer	Mount count
NET	string	TRUE if network device



Item	Data Type	Information Returned
NEXTDEVNAM	string	Device name of next volume in volume set
ODV	string	TRUE if device is capable of output
OPCNT	integer	Operation count
OPR	string	TRUE if device is an operator
OWNUIC	integer	UIC of device owner
PID	string	Process identification of device owner
RCK	string	TRUE if device has read checking enabled
REC	string	TRUE if device is record oriented
RECSIZ	integer	Blocked record size
RND	string	TRUE if device allows random access
ROOTDEVNAM	string	Device name of root volume in volume set
RTM	string	TRUE if device is real-time
SDI	string	TRUE if device is single directory structured
SECTORS	integer	Number of sectors per track
SERIALNUM	integer	Volume serial number
SHR	string	TRUE if device is shareable
SPL	string	TRUE if device is spooled
SPLDEVNAM	string	Spooled device name
SQD	string	TRUE if device is sequential block oriented
STS	integer	Status information
SWL	string	TRUE if device is software write locked
TRACKS	integer	Number of tracks per cylinder
TRANSCNT	integer	Volume transaction count
TRM	string	TRUE if device is a terminal
TT_ALTYPEAHD	string	TRUE if terminal has an alternate type ahead buffer
TT_ANSICRT	string	TRUE if terminal is an ANSI CRT terminal
TT_APP_KEYPAD	string	TRUE if terminal keypad is in applications mode
TT_AUTOBAUD	string	TRUE if terminal has automatic baud rate detection
TT_AVO	string	TRUE if terminal has a VT100-family display
TT_BLOCK	string	TRUE if the terminal has block mode capability
TT_BRDCSTMBX	string	TRUE if terminal uses mailbox broadcast messages

**LEX-16    Lexical Functions**  
**F\$GETDVI**

Item	Data Type	Information Returned
TT_CRFILL	string	TRUE if terminal requires fill after RET
TT_DECCRT	string	TRUE if terminal is a DIGITAL CRT terminal
TT_DIALUP	string	TRUE if terminal is connected to dialup
TT_DISCONNECT	string	TRUE if terminal can be disconnected
TT_DMA	string	TRUE if the terminal has DMA mode
TT_DRCS	string	TRUE if terminal supports loadable character font
TT_EDIT	string	TRUE if terminal edit characteristic is set
TT_EDITING	string	TRUE if terminal advanced editing is enabled
TT_EIGHTBIT	string	TRUE if terminal uses 8-bit ASCII character set
TT_ESCAPE	string	TRUE if terminal generates escape sequences
TT_FALLBACK	string	TRUE if terminal uses multinational fallback option
TT_HALFDUP	string	TRUE if terminal is in half-duplex mode
TT_HANGUP	string	TRUE if terminal has hangup characteristic set
TT_HOSTSYNC	string	TRUE if terminal has host/terminal communication
TT_INSERT	string	TRUE if insert-mode is the default line-editing mode for terminal
TT_LFFILL	string	TRUE if terminal requires fill after LF
TT_LOCALECHO	string	TRUE if terminal has local echo characteristic set
TT_LOWER	string	TRUE if terminal has lowercase characters set
TT_MBXDSABL	string	TRUE if mailboxes associated with the terminal will receive unsolicited input notification or input notification
TT_MECHFORM	string	TRUE if terminal has mechanical form feed
TT_MECHTAB	string	TRUE if terminal has mechanical tabs and is capable of tab expansion
TT_MODEM	string	TRUE if terminal is connected to a modem
TT_MODHANGUP	string	TRUE if terminal has modify hang-up characteristic set
TT_NOBRDCST	string	TRUE if terminal will receive broadcast messages
TT_NOECHO	string	TRUE if terminal does not echo input characters
TT_NOTYPEAHD	string	TRUE if data must be solicited by a read operation

## Lexical Functions    LEX-17

### F\$GETDVI

Item	Data Type	Information Returned
TT_OPER	string	TRUE if terminal is an operator terminal
TT_PASTHRU	string	TRUE if terminal has passall with flow control
TT_PRINTER	string	TRUE if terminal has available printerport
TT_READSYNC	string	TRUE if terminal has read synchronization
TT_REGIS	string	TRUE if terminal has REGIS graphics
TT_SCOPE	string	TRUE if terminal has a video screen display
TT_SECURE	string	TRUE if terminal can recognize the secure server
TT_SETSPEED	string	TRUE if you can set the speed on the terminal line
TT_SIXEL	string	TRUE if the sixel is supported
TT_SYSPWD	string	TRUE if the system password is enabled for a particular terminal
TT_TTSYNC	string	TRUE if terminal/host synchronization exists
TT_WRAP	string	TRUE if a new line is inserted when the cursor moves beyond the right margin
UNIT	integer	Unit number
VOLCOUNT	integer	Volumes in volume set
VOLNAM	string	Volume name
VOLNUMBER	integer	Current volume in volume set
VOLSETMEM	string	TRUE if disk is in a volume set
VPROT	string	Volume protection mask
WCK	integer	TRUE if write checking is enabled

The following example returns the error count for DUA0.

```
$ ERR = F$GETDVI ("DUA0","ERRCNT")
$ SHOW SYMBOL ERR
ERR = 0    Hex = 00000000    Octal = 000000
```

## LEX-18 Lexical Functions

### F\$GETJPI

#### F\$GETJPI (process-id,item)

Returns a specified item of information about a specified process.

#### ARGUMENTS

##### process-id

A character string containing a process identification number (leading zeros can be omitted). A null string ("") or a numeric zero identifies the current process.

##### item

A character string containing one of the following (do not abbreviate):

Item	Data	Information Returned
ACCOUNT	string	Account name string (8 characters filled with trailing blanks)
APTCNT	integer	Active page table count
ASTACT	integer	Access modes with active ASTs
ASTCNT	integer	Remaining AST quota
ASTEN	integer	Access modes with ASTs enabled
ASTLM	integer	AST limit quota
AUTHPR	integer	Maximum priority that a process without the ALTPRI privilege can achieve
AUTHPRIV	string	Privileges that a process can enable
BIOCNT	integer	Remaining buffered I/O quota
BIOLM	integer	Buffered I/O limit quota
BUFIO	integer	Count of process-buffered I/O operations
BYTCNT	integer	Remaining buffered I/O byte-count quota
BYTLM	integer	Buffered I/O byte-count limit quota
CLINAME	string	Current command language interpreter; always returns "DCL"
CPULIM	integer	Limit on process CPU time
CPUTIM	integer	CPU time used in hundredths of a second
CURPRIV	string	Current process privileges
DFPFC	integer	Default page fault cluster size
DFWSCNT	integer	Default working set size
DIOCNT	integer	Remaining direct I/O quota
DIOLM	integer	Direct I/O limit quota
DIRIO	integer	Count of direct I/O operations for the process
EFCS	integer	Local event flags 0 through 31
EFCU	integer	Local event flags 32 through 63

Item	Data	Information Returned
EFWM	integer	Event flag wait mask
ENQCNT	integer	Lock request quota remaining
ENQLM	integer	Lock request quota limit
EXCVEC	integer	Address of a list of exception vectors
FILCNT	integer	Remaining open file quota
FILLM	integer	Open file quota
FINALEXC	integer	Address of a list of final exception vectors
FREPOVA	integer	First free page at end of program region
FREPIVA	integer	First free page at end of control region
FREPTECNT	integer	Number of pages available for virtual memory expansion
GPGCNT	integer	Global page count in working set GRP
IMAGECOUNT	integer	Number of images that have been run down for the process
IMAGNAME	string	File name of the current image
IMAGPRIV	string	Privileges with which the current image was installed
JOBRCCNT	integer	Number of subprocesses owned by the process
LOGINTIM	string	Process creation time
MASTER_PID	string	Returns the process identification of the process at the top of the current job's process tree.
MEM	integer	Member number of UIC
MODE	string	Process mode (BATCH, INTERACTIVE, NETWORK, or OTHER)
MSGMASK	integer	Default message mask
OWNER	string	Process identification of process owner
PAGEFLTS	integer	Count of page faults
PAGFILCNT	integer	Remaining paging file quota
PAGFILLOC	integer	Location of the paging file
PGFLQUOTA	integer	Paging file quota (maximum virtual page count)
PHDFLAGS	integer	Flags word
PID	string	Process identification
PPGCNT	integer	Process page count
PRCCNT	integer	Count of subprocesses
PRCLM	integer	Subprocess quota
PRCNAM	string	Process name
PRIB	integer	Process's base priority
PROCPRIV	integer	Process's default privileges

## LEX-20 Lexical Functions

### F\$GETJPI

Item	Data	Information Returned
SITESPEC	integer	Per-process site-specific longword
STATE	string	Process state
STS	integer	Process status flags
SWPFILLOC	integer	Location of the swap file
TERMINAL	string	Login terminal name for interactive users (1-7 characters)
TMBU	integer	Termination mailbox unit number
TQCNT	integer	Remaining timer queue entry quota
TQLM	integer	Timer queue entry quota
UIC	string	Process's UIC
USERNAME	string	User name string
VIRTPEAK	integer	Peak virtual address size
VOLUMES	integer	Count of currently mounted volumes
WSAUTH	integer	Maximum authorized working set size
WSAUTHEXT	integer	Maximum authorized working set extent
WSEXTENT	integer	Current working set extent
WSPEAK	integer	Working set peak
WSQUOTA	integer	Working set size quota
WSSIZE	integer	Process's current working set size

The following example returns the user name for process 003B0018.

```
$ NAME = F$GETJPI ("3B0018", "USERNAME")
$ SHOW SYMBOL NAME
NAME = "USER"
```

### F\$GETSYI (item,[node])

Returns a specified item of information about the system.

#### ARGUMENTS

##### item

A character string containing one of the following (do not abbreviate):

Item	Information Returned and Data Type
ARCHFLAG	Architecture flags (string)
BOOTTIME	Time the system was booted (string)



Item	Information Returned and Data Type
CHARACTER__ EMULATED	TRUE if the character string instructions are emulated on the CPU (string)
CLUSTER__MEMBER	TRUE if the node is a member of a cluster; CLUSTER__MEMBER always returns FALSE on a MicroVMS system (string)
CPU	Processor type: 7 for a MicroVAX I (integer)
DECIMAL__ EMULATED	TRUE if the decimal string instructions are emulated on the CPU (string)
D__FLOAT__ EMULATED	TRUE if the D__floating instructions are emulated on the CPU (string)
F__FLOAT__ EMULATED	TRUE if the F__floating instructions are emulated on the CPU (string)
G__FLOAT__ EMULATED	TRUE if the G__floating instructions are emulated on the CPU (string)
PAGEFILE__FREE	Returns the number of free pages in the currently installed paging files (integer)
PAGEFILE__PAGE	Returns the number of pages in the currently installed paging files (integer)
SID	System identification (integer)
SWAPFILE__FREE	Returns the number of free pages in the currently installed swapping files (integer)
SWAPFILE__PAGE	Returns the number of pages in the currently installed swapping files (integer)
VERSION	Version of VMS in use (string)

The following example returns the system identification.

```
$ SID = F$GETSYI ("SID")
$ SHOW SYMBOL SID
   SID = 117441280   Hex = 070003000   Octal = 0700001400
```

## **F\$IDENTIFIER (identifier,translation)**

Converts an alphanumeric UIC to its numeric equivalent, or a numeric UIC to its alphanumeric equivalent. (Alphanumeric UICs are optionally equated to numeric UICs in the rights database.)

### **ARGUMENTS**

#### **identifier**

A character string containing an identifier.

## LEX-22 Lexical Functions

### F\$IDENTIFIER

#### translation

If *identifier* is alphanumeric, specify *translation* as a character string containing NAME\_TO\_NUMBER. If *identifier* is numeric, specify *translation* as a character string containing NUMBER\_TO\_NAME.

The following example translates the alphanumeric identifier JONES to its numeric equivalent.

```
$ NAME = F$IDENTIFIER("JONES", "NAME_TO_NUMBER")
$ SHOW SYMBOL NAME
Name = 589926 Hex = 00090066 Octal = 00002200146
```

### F\$INTEGER (string)

Converts the specified string to a number. The character string must be null (zero) or a valid decimal specification (no hexadecimal or octal specifications) of a positive or negative integer. If the character string does not contain a numeric character string, it is converted to either a 1 (if the first character is Y,y,T, or t) or a 0.

#### ARGUMENTS

##### string

A character string or integer.

The following example (which would appear in a command procedure) reads a character string from the terminal and converts it to a number.

```
$ INQUIRE DOG_COUNT "Number of dogs"
$ DOG_COUNT = F$INTEGER (DOG_COUNT)
```

### F\$LENGTH (string)

Returns the length of the specified character string.

#### ARGUMENTS

##### string

A character string whose length is being determined.

The following example indicates that the character string equated to REPORT\_1 is 56 characters in length.

```
$ LINE_LENGTH = F$LENGTH (REPORT_1)
$ SHOW SYMBOL LINE_LENGTH
LINE_LENGTH = 56 Hex = 00000038 Octal = 000070
```

## F\$LOCATE (substring,string)

Locates a specified portion of a character string and returns as a number the offset of the first character. The first character in a string is always offset position 0 from the beginning of the string. If the substring is not present, the offset of the last character in the character string plus one is returned.

### ARGUMENTS

#### substring

The character string in *string* that you want to locate.

#### string

A character string to be edited by F\$LOCATE.

The following example returns the position of the substring DOGS.

```
$ SHOW SYMBOL LINE_1
  LINE_1 = "Number of dogs:"
$ DOGS = F$LOCATE ("dogs",LINE_1)
$ SHOW SYMBOL DOGS
  DOGS = 10  Hex = 0000000A  Octal = 12
```

## F\$MESSAGE (message-code)

Returns as a character string the facility, severity, identification and text associated with the specified message code.

### ARGUMENTS

#### message-code

An integer for which you are requesting error message text. The message code must be specified as a number (decimal, hexadecimal, or octal).

The following example returns the success message.

```
$ MESS = F$MESSAGE (1)
$ SHOW SYMBOL MESS
  MESS = "%SYSTEM-S-NORMAL, normal successful completion"
```

## LEX-24 Lexical Functions

### F\$MODE

#### F\$MODE ( )

Returns the character string INTERACTIVE if you issue the command from the terminal, the character string BATCH if you issue the command from a batch job, and the character string NETWORK if you issue the command from a network job. In the following example, you issue the command from the terminal.

```
$ MODE = F$MODE ( )
$ SHOW SYMBOL MODE
MODE = "INTERACTIVE"
```

#### F\$PARSE (file-spec,[def-spec1],[def-spec2],[field,...],[type])

Returns either the full file specification for the specified file or one of the following fields: NODE, DEVICE, DIRECTORY, NAME, TYPE, VERSION. The device name in the resulting file specification must be a valid device. The directory must be a valid directory on that device. An error in the resulting file specification returns a null string (unless the SYNTAX\_ONLY type is specified or a specific field is requested). Logical names and device names must terminate with a colon. If you omit arguments, commas to the left of the last specified argument must be included as place holders.

#### ARGUMENTS

##### file-spec

A character string containing a file specification. If *file-spec* is not a full file specification, defaults are taken from (1) your current default directory if *def-spec1* is not specified; (2) *def-spec1* and then your current default directory if *file-spec* and *def-spec1* are specified; (3) *def-spec1*, then *def-spec2*, and then your current default directory if both *def-spec1* and *def-spec2* are specified. The file name, file type, and version number are null if not specified in either *file-spec* or *def-spec1*. Wildcards can be used.

##### def-spec1

A character string which is substituted in the output string if a particular field in *file-spec* is missing.

##### def-spec2

A character string which is substituted in the output string if a particular field in *file-spec* and *def-spec1* is missing.

##### field

A character string containing one of the following fields (do not abbreviate field names): NODE, DEVICE, DIRECTORY, NAME, TYPE, VERSION.

### type

The type of parsing to be done. Valid types are:

SYNTAX_ONLY	Does not check for the existence of the directory or device
NO_CONCEAL	Ignores the "conceal" attribute in the translation of a logical name as part of the file specification

The following example returns the directory name for the file 1983.DAT in your default directory.

```
$ D = F$PARSE ("1983.DAT" , , "DIRECTORY")
$ SHOW SYMBOL D
D = "[ACCOUNTS]"
```

The next example returns the full file specification.

```
$ D = F$PARSE ("1983.DAT")
$ SHOW SYMBOL D
D = "DUA1:[ACCOUNTS]1983.DAT;"
```

The next example uses DOG subdirectory as a default directory.

```
$ D = F$PARSE ("1983.DAT" , "DUA1:[LICENSES.DOG]")
$ SHOW SYMBOL D
D = "DUA1:[LICENSES.DOG]1983.DAT;"
```

The device name in the resulting file specification must be a valid device. The directory name must be a valid directory on that device. An error in the resulting file specification returns a null string.

### F\$PID (context-symbol)

Returns a process identification (PID) number, and updates the context symbol to identify the current position in the system's process list. If context-symbol is equated to zero or a null string (""), returns as a character string the process identification number of the first process on the system. If context-symbol was used in a previous F\$PID function and has not been redefined, F\$PID returns the process identification number of the next process on the system. A null string is returned after the last process on the system has been examined. If your system has individual accounts, you may not have access to all processes on the system.

#### ARGUMENTS

##### context\_symbol

A symbol that DCL uses to store a pointer into the system's list of processes. The first time you use F\$PID, use a symbol that is undefined or equated to the null string ("").

## LEX-26 Lexical Functions

### F\$PID

The following command procedure returns the process identification number of each process on the system.

```
$ ! SYSPID.COM
$ !
$ NEXT = 0
$ START:
$ PID = F$PID (NEXT)
$ IF PID .EQS. "" THEN EXIT
$ SHOW SYMBOL PID
$ GOTO START
```

### F\$PRIVILEGE (priv-list)

Returns a string containing TRUE or FALSE, depending on whether all of the privileges are set as specified. You can specify either the positive or negative version of a privilege.

#### ARGUMENTS

##### priv-list

A character string identifying a privilege or a list of privileges separated by commas.

The following example finds that the current process does not have one SYSPRV, does not have WORLD, or does not have either. To determine which privileges are not set, you would have to check each individually.

```
$ PRIVS = F$PRIVILEGE ("SYSPRV,WORLD")
$ SHOW SYMBOL PRIVS
PRIVS = "FALSE"
```

### F\$PROCESS ( )

Returns as a character string the name of the current process, as demonstrated in the following example.

```
$ PROC = F$PROCESS ( )
$ SHOW SYMBOL PROC
PROC = "USER"
```



## F\$SEARCH (file-spec,[stream-id])

Returns the full file specification of *file-spec*. If device or directory are omitted, the current defaults are used. If version is omitted, the latest version number is used. If *file-spec* contains wildcards, each time F\$SEARCH is called, the next file specification that agrees with *file-spec* is returned. A null string is returned after the last file specification that agrees with *file-spec*.

### ARGUMENTS

#### file-spec

The file name and type of the file specification to be searched for; wildcards are permitted.

#### stream-id

A positive integer used to maintain separate search contexts.

The following command procedure returns the latest versions of all the type EXE files in the SYS\$SYSTEM directory.

```
$ ! SEARCH.COM
$ !
$ START:
$ FILE = F$SEARCH ("SYS$SYSTEM:*.EXE")
$ IF FILE .EQS. "" THEN EXIT
$ SHOW SYMBOL FILE
$ GOTO START
```

The following command procedure returns all COM and DAT files.

```
$ ! COM_DAT.COM
$ !
$ ! find first COM and DOC files
$ COM = F$SEARCH ("*.COM;*",1)
$ DOC = F$SEARCH ("*.DOC;*",2)
$ START:
$     ! set DONE to true
$     DONE = 1
$
$     ! search for COM file
$     IF (COM .EQS. "") THEN GOTO END_COM
$     SHOW SYM COM
$     COM = F$SEARCH ("*.COM;*",1)
$     DONE = 2           ! DONE is false
$     END_COM:
$
```

## LEX-28 Lexical Functions

### F\$SEARCH

```
$      ! search for DOC file
$      IF (DOC .EQS. "") THEN GOTO END_DOC
$          SHOW SYM DOC
$          DOC = F$SEARCH ("*.DOC;*",2)
$          DONE = 2          ! DONE is false
$          END_DOC:
$
$
$      ! if no DOC or COM files, DONE is still true
$      IF DONE THEN EXIT
$      GOTO START
```

### F\$SETPRV (priv-list)

If the process has the proper authorization, F\$SETPRV enables or disables the specified privileges. In addition, F\$SETPRV returns a string containing the state of the specified privileges before F\$SETPRV was executed.

#### ARGUMENTS

##### priv-list

A character string defining a privilege or list of privileges separated by commas. You can specify either the positive or negative version of a privilege.

The following example sets SYSPRV and WORLD. The current process had SYSPRV; WORLD was set by F\$SETPRV.

```
$ PRIV = F$SETPRV ("WORLD,SYSPRV")
$ SHOW SYMBOL PRIV
PRIV = "NOWORLD,SYSPRV"
$ TEST = F$PRIVILEGE ("WORLD,SYSPRV")
TEST = "TRUE"
```

### F\$STRING (expression)

Returns the string that is equivalent to the specified expression.

#### ARGUMENTS

##### expression

An integer or string expression. When converting an integer to a string, F\$STRING uses decimal notation, omitting leading zeros. If the integer is negative, a minus sign precedes the number.

In the following example, F\$STRING evaluates the integer expression and converts it to a character string.

```
$ DOG_1 = 12
$ DOG_2 = 13
$ TOTAL_DOGS = F$STRING (DOG_1 + DOG_2)
$ SHOW SYMBOL TOTAL_DOGS
TOTAL_DOGS = "25"
```

## F\$TIME ( )

Returns as a character string the current date and time in absolute time format, as demonstrated in the following example.

```
$ TIME = F$TIME ( )
$ SHOW SYMBOL TIME
TIME = "21-OCT-1983 12:45:22"
```

## F\$TRNLNM (name,[table],[index],[mode],[case],[item])

Finds a logical name and returns the translation or the requested attributes. If the logical name does not exist, F\$TRNLNM returns a null string. (Section 2.3 discusses logical names.) If you omit arguments, commas to the left of the last specified argument must be included as place holders.

### ARGUMENTS

#### name

A character string containing the logical name.

#### table

A character string containing a logical name whose equivalence names are the logical name tables to be searched (the search order is determined when the logical name is defined). The *table* argument defaults to LNM\$DEFAULT\_SEARCH whose equivalence names are process, job, group, and system logical name tables, in that order.

#### index

The number of the equivalence name to return if the logical name has more than one translation; defaults to 0.

#### mode

A character string containing one of the following access modes (do not abbreviate): USER (default), SUPERVISOR, EXECUTIVE, or KERNEL.

## LEX-30 Lexical Functions

### F\$TRNLNM

#### case

A character string containing one of the following (do not abbreviate):  
CASE\_BLIND (F\$TRNLNM searches for a logical name match without taking into account character casing; default) or CASE\_SENSITIVE (F\$TRNLNM searches for an exact logical name match).

#### item

A character string containing one of the following (do not abbreviate):

Item	Data Type	Information Returned
ACCESS_MODE	string	Access mode associated with the logical name
CONCEALED	integer	TRUE if the logical name is concealed
CONFINED	string	TRUE if the logical name is confined
LENGTH	integer	Length of the equivalence name
MAX_INDEX	integer	Number of logical name translations
NO_ALIAS	string	TRUE if the logical name must be unique within its access mode and outer access modes
TABLE	string	TRUE if the logical name is the name of a logical name table
TABLE_NAME	string	Name of the table containing the logical name
TERMINAL	string	TRUE if the logical name cannot be translated iteratively
VALUE	string	Default. Equivalence name

In the following example, the logical name GROUP\_NAMES is confined (it is not copied to subprocesses).

```
$ CONFINE = F$TRNLNM ("GROUP_NAMES", , , , "CONFINE")
$ SHOW SYMBOL CONFINE
CONFINE = "TRUE"
```

#### F\$TYPE (symbol)

Returns the data type of the symbol. If *symbol* forms a valid integer, F\$TYPE returns INTEGER. If *symbol* is a character string whose characters do not form a valid integer, F\$TYPE returns STRING. If *symbol* is undefined, F\$TYPE returns a null string.

## ARGUMENTS

### **symbol**

A character string or integer which references the name of the symbol to be edited.

The following example returns the data type INTEGER.

```
$ NUM = "673"  
$ TYPE = F$TYPE (NUM)  
$ SHOW SYMBOL TYPE  
  TYPE = "INTEGER"
```

### **F\$USER ( )**

Returns as a character string the current UIC, as demonstrated in the following example.

```
$ UIC = F$USER (  
$ SHOW SYMBOL UIC  
  UIC = "[DEVELOPMENT,OSGOOD]"
```

### **F\$VERIFY ([procedure],[image])**

Returns a value indicating whether the procedure verification setting is on or off. If called with the first argument, turns procedure and image verification on if the argument is 1 and off if the argument is 0. If called with both arguments, turns procedure verification on or off depending on the value of the first argument and turns image verification on or off depending on the value of the second argument. If you specify only the second argument, you must include the comma as a place holder.

## ARGUMENTS

### **procedure**

An integer with a value of 1 to turn procedure verification on or a value of 0 to turn procedure verification off.

### **image**

An integer with a value of 1 to turn procedure verification on or a value of 0 to turn procedure verification off.

## LEX-32    Lexical Functions

### F\$VERIFY

The following command procedure saves the user's setting of procedure and image verification at the beginning of the procedure and then restores those settings at the end of the procedure.

```
$ ! PROCEDURE.COM
$ !
$ SAVE_PROC_VER = F$ENVIRONMENT ("VERIFY_PROCEDURE")
$ SAVE_IMAGE_VER = F$ENVIRONMENT ("VERIFY_IMAGE")
.
.   ! commands
.
$ GARBAGE = F$VERIFY (SAVE_PROC_VER, SAVE_IMAGE_VER)
```



# Appendix MAIL

## MAIL

The DCL command MAIL sends a specified file to a specified user or users. Typing the MAIL command without a file specification and user name invokes the interactive Mail Utility.

### MAIL.1 DCL Command

Use the DCL command MAIL with parameters and the optional /SELF and /SUBJECT qualifiers to send a file to other users without invoking the interactive Mail Utility. Use the DCL command MAIL without parameters and with the optional /EDIT qualifier to invoke the interactive Mail Utility.

#### MAIL [file-spec] [username,...]

When the optional parameters are specified, MAIL sends a file to the specified users. When the optional parameters are not specified, MAIL invokes the interactive Mail Utility.

#### PARAMETERS

##### **file-spec**

Specification of the file to be sent. No wildcard characters are allowed. The file type defaults to TXT.

##### **username**

Name of the user or distribution list to receive the mail message. If you include a distribution list, precede the name of the list with an at sign (@), and enclose both the at sign and the name in quotation marks.

## MAIL-2 MAIL MAIL

### QUALIFIERS

**/EDIT[=(keyword,...)]**

**/NOEDIT**

Invokes EDT for the Mail Utility commands FORWARD, REPLY, or SEND used during the current interactive session. If you do not specify a keyword, the default is MAIL/EDIT=(SEND,REPLY). Possible keywords are:

FORWARD	Invokes EDT to edit the last message read before sending it to the specified users.
REPLY[=EXTRACT]	Invokes EDT to edit your response to the last message read. Specify the keyword EXTRACT to edit the message to which you are replying.
SEND	Invokes EDT to edit the message you are sending.

**/SELF**

**/NOSELF**

Sends you a copy of the file you are sending. When you send a message from the DCL command level (instead of from within the Mail Utility), /SELF or /NOSELF overrides any setting you have established with the SET COPY\_SELF command.

**/SUBJECT="text"**

Specifies the subject of the mail. Enclose *text* in quotation marks if the subject consists of more than one word.

### EXAMPLES

**\$ MAIL/SUBJECT="PROJECT DAWN" DAWN.DAT BRUTUS::OSGOODE, PORTIA::RIPLEY**

Sends the file DAWN.DAT to user Osgoode on node BRUTUS and to user Ripley on node PORTIA. The subject of the message is PROJECT DAWN.

**\$ MAIL DAWN.DAT BRUTUS::OSGOODE, "@ALLBUD"**

Sends the file DAWN.DAT to user Osgoode on node BRUTUS and to everyone on the distribution list ALLBUD.DIS.

**\$ MAIL/NOSELF MYFILE.DAT CHARLES**

Sends the file MYFILE.DAT to user CHARLES without sending a copy back to you.

**\$ MAIL**

Invokes the MAIL Utility.

## **MAIL.2 Interactive Mail Utility**

The Mail Utility allows you to send, receive, and manipulate messages either by typing MAIL commands or by entering them on the default MAIL keypad.

### **MAIL.2.1 MAIL Commands**

Type MAIL commands in response to the MAIL prompt.

#### **ANSWER [file-spec]**

Sends a response to the message you have just read. If you include a file specification, the specified file is sent as your answer; otherwise, you will be prompted for the text of your message. (ANSWER is interchangeable with REPLY.)

##### **PARAMETERS**

##### **file-spec**

The specification of the file to be sent as a reply.

##### **QUALIFIERS**

##### **/EDIT**

##### **/NOEDIT (default)**

Invokes the editor to create or edit the message you are sending. The EXIT command completes the ANSWER operation; the QUIT command cancels the ANSWER operation. The /NOEDIT qualifier overrides the ANSWER/EDIT default established with the DCL command MAIL/EDIT.

##### **/EXTRACT**

Enables you to use the editor to edit the message to which you are replying rather than starting a new message.

##### **/LAST**

Inserts the message most recently sent as text for the reply. The /LAST qualifier cannot be used with any of the other qualifiers for the ANSWER command or with a file specification.

##### **/SELF**

##### **/NOSELF (default)**

Determines whether or not MAIL sends you a copy of your response. The default is /NOSELF, unless you have used the SET COPY\_SELF command to specify that copies be sent to you automatically.

## MAIL-4    MAIL ATTACH

### ATTACH process name

Transfers control of your terminal from your current process (which then hibernates) to the specified process. (Note that you always SPAWN to a new process and ATTACH to a process that already exists.)

#### PARAMETERS

##### **process-name**

The name of the process or subprocess to which the connection is to be made.

#### QUALIFIERS

##### **/PARENT**

Indicates that you want to attach to the parent process of your current process.

You cannot specify the process-name parameter with the /PARENT qualifier.

#### EXAMPLE

```
$ SPAWN MAIL
```

```
%DCL-S-SPAWNED, process OSGOODE_1 spawned
```

```
%DCL-S-ATTACHED, terminal now attached to process OSGOODE_1
```

```
MAIL> DIRECTORY MAIL
```

```
.
```

```
.
```

```
.
```

```
MAIL> ATTACH OSGOODE
```

```
%DCL-S-RETURNED, control returned to process OSGOODE
```

```
.
```

```
.
```

```
.
```

```
CTRL/Y
```

```
$ ATTACH OSGOODE_1
```

Enters the DCL command SPAWN to create a subprocess (OSGOODE\_1), which invokes MAIL and uses the ATTACH command to move between MAIL (OSGOODE\_1) and the DCL command level (OSGOODE). The ATTACH command allows you to transfer control between processes.

### BACK

Displays the previous message if the last command issued was READ. When the last command issued was the DIRECTORY command, the BACK command displays the previous screen of the directory display.

## QUALIFIERS

### **/EDIT**

Invokes the editor. You can use the editor to peruse the previous message (enter the QUIT command when finished) or you can edit the previous message and save the new version in a sequential file (when finished, enter the EXIT command and supply a file name).

## **COMPRESS [file-spec]**

Makes an ISAM mail file smaller. When you compress a file, the following four steps occur:

1. A temporary file named MAIL\_####\_COMPRESS.TMP is created (#### is a unique four-digit number).
2. The contents of the file to be compressed are copied to the temporary file and compressed.
3. The original (uncompressed) file is renamed with a file type of OLD.
4. The newly compressed file is renamed from MAIL\_####\_COMPRESS.TMP to its original name.

## PARAMETERS

### **file-spec**

The name of the mail file to be compressed. If the name of a file is not specified, MAIL will compress the mail file that is currently open. If there is no open mail file, MAIL will compress the default mail file, MAIL.MAI.

## QUALIFIERS

### **/OUTPUT=output-file-spec**

Specifies the name of the compressed file.

## EXAMPLE

```
MAIL> COMPRESS SOCIAL.MAI
%MAIL-S-CREATED, WORKDISK:[OSGOODE]MAIL_08C8_COMPRESS.TMP;1 created
%MAIL-S-COPIED, WORKDISK:[OSGOODE]SOCIAL.MAI;1 copied to
WORKDISK:[OSGOODE]
MAIL_08C8_COMPRESS.TMP;1 (2 records)
%MAIL-S-RENAMED, WORKDISK:[OSGOODE]SOCIAL.MAI;1 renamed to
WORKDISK:[OSGOODE]
SOCIAL.OLD;2
%MAIL-S-RENAMED, WORKDISK:[OSGOODE]MAIL_08C8_COMPRESS.TMP;1 renamed to
WORKDISK:[OSGOODE]SOCIAL.MAI;1
```

Compresses the contents of a file named SOCIAL.MAI.

**MAIL-6    MAIL**  
**COPY**

**COPY** foldername [filename]

Copies a message to another folder without deleting it from the current folder. If the specified folder does not exist, it is created. (Enter CTRL/C to cancel the operation without exiting from MAIL.)

**PARAMETERS**

**foldername**

The name of the folder to which the message is to be copied. If the specified folder does not exist (and you have not entered the qualifier /NOCONFIRM) you are asked whether you want to create it. A folder name can be 1 to 39 characters in length. Valid characters for folder names are A through Z, a through z, \$, —, and 0 through 9.

**filename**

The name of the mail file to which the message is copied. The default is the current mail file. If the specified mail file does not exist, it is created.

**QUALIFIERS**

**/ALL**

Copies all the currently selected messages to another folder. (See the SELECT command for more information on selecting messages.)

**/CONFIRM (default)**

**/NOCONFIRM**

Determines whether or not you will be queried about creating a new folder.

**EXAMPLE**

MAIL> 2

.  
.  
.

MAIL> COPY

\_Folder: MEMOS

\_File: RETURN

Folder MEMOS does not exist.

Do you want to create it (Y/N, default is N)? Y

%MAIL-I-NEWFOLDER, folder MEMOS created

Puts a copy of mail message #2 into the new folder MEMOS in the default mail file.



## CURRENT

Displays the beginning of the message you are currently reading.

### QUALIFIERS

#### /EDIT

Invokes the editor. You can invoke the editor to peruse the current message (enter the QUIT command when finished) or you can edit the current message and save the new version in a sequential file (enter the EXIT command and supply a file name).

## DEFINE/KEY key-name string

Defines a key to execute a MAIL command. Defining a key enables you to press that key to enter a command instead of typing the command name. For information on the MAIL keypad, see Section MAIL.3.

To increase the number of key definitions available on your terminal, you can use the /SET\_STATE qualifier. The same key can be assigned any number of definitions as long as each definition is associated with a different key state. State names can be any alphanumeric string (for example, GOLD, SWITCH, SNAG1, or SNAG2).

Any key definitions you define during a MAIL session will disappear when you EXIT from the Mail Utility. To retain keypad key definitions from one MAIL session to another, create a file in your top-level directory with a file type of INI containing these definitions (for example, MAIL\_KEYDEF.INI). Enter the following command in your login command file (LOGIN.COM):

```
$ DEFINE MAIL$INIT SYS$LOGIN:MAIL_KEYDEF.INI
```

The file you create (MAIL\_KEYDEF.INI) containing your key definitions will act as a login command file for MAIL.

### PARAMETERS

#### key-name

The name of the key you are defining.

The following table lists the key names to use when you define keys:

Keyname	VT100 Key	VT200 Key
PF1	PF1	PF1
PF2	PF2	PF2
PF3	PF3	PF3

## MAIL-8 MAIL DEFINE/KEY

Keyname	VT100 Key	VT200 Key
PF4	PF4	PF4
KP0, KP1- KP9	Keypad 0- 9	Keypad 0- 9
PERIOD	Period key	Period key
COMMA	Comma key	Comma key
MINUS	Minus key	Minus key
ENTER	Enter key	Enter key
FIND, INSERT HERE	—	Find, Insert Here
REMOVE, SELECT	—	Remove, Select
PREV_SCREEN	—	Prev Screen
NEXT_SCREEN	—	Next Screen
HELP, DO	—	Help(F15), Do(F16)
F17- F20	—	Function Keys F17- F20

(Note that you cannot define the arrow keys or function keys F1 through F14.) The keys PF1- PF4, KP0- KP9, PERIOD, COMMA, MINUS, and ENTER all have default definitions. You may, however, override those definitions with your own key definitions.

### PARAMETERS

#### string

The MAIL command string to be entered when the specified key is pressed.

### QUALIFIERS

#### /ECHO (default)

#### /NOECHO

Specifies whether or not the command line is echoed after you press the defined key. You cannot define a key specifying both /NOECHO and /NOTERMINATE.

#### /IF\_STATE=state-list

#### /NOIF\_STATE=state-list (default)

Specifies a list of one or more states, one of which must be set in order to enable the specified key definition. If you omit or negate this qualifier, the default is the current state.

#### /LOCK\_STATE

#### /NOLOCK\_STATE (default)

Retains the state specified by the /SET\_STATE qualifier until you use the /SET\_STATE qualifier again to change it.

**/LOG (default)**  
**/NOLOG**

Specifies whether or not informational messages that signal successfully created key definitions are displayed.

**/SET\_STATE=state-name**  
**/NOSET\_STATE=state-name (default)**

Associates a state with the key you are defining. A state name can be any alphanumeric string. If you omit or negate this qualifier, the current state remains unchanged. You cannot define a key specifying both /SET\_STATE and /TERMINATE.

**/TERMINATE**  
**/NOTERMINATE (default)**

Determines whether or not the specified command string executes when you press the key. When you use /NOTERMINATE you must press RETURN to execute the command string. You cannot define a key specifying both /SET\_STATE and /TERMINATE.

**EXAMPLE**

MAIL> DEFINE/KEY PF4 "SET " /SET\_STATE=SUB2

MAIL> DEFINE/KEY KP3 "FOLDER" /ECHO/TERMINATE/IF\_STATE=SUB2

The first command defines the PF4 keypad key as the SET command and associates this key with the state named SUB2. The second command defines keypad key 3 to be "FOLDER" and makes it dependent on a state named SUB2. When you press the PF4 key followed by keypad key 3, MAIL executes the SET FOLDER command.

**DELETE [message-number]**

Moves the specified message to the WASTEBASKET folder. The message is not actually deleted until you exit from MAIL or read another message file. If you enter QUIT or CTRL/Y before exiting from MAIL or before reading another message, the delete operation is canceled. To recover a message from the WASTEBASKET folder, SELECT the WASTEBASKET folder, READ the desired message, and MOVE it to another folder.

**PARAMETERS**

**message-number**

The number of the message that is to be deleted from the current folder. Defaults to the most recently read mail message.

## **MAIL-10    MAIL DELETE**

### **QUALIFIERS**

#### **/ALL**

Deletes all the messages in the currently selected folder (see the SELECT command for more information).

#### **EXAMPLE**

MAIL> **DELETE 2**

Deletes message number 2 from the current folder.

### **DIRECTORY [foldername]**

Displays a list of the currently selected messages in the specified folder, listing message number, sender's name and node, date, and subject. If there are no currently selected messages, MAIL displays a directory of the messages in the NEWMAIL folder (if any unread messages exist) or the MAIL folder. (See the SELECT command for more information about selecting messages.)

#### **PARAMETERS**

##### **foldername**

The name of the folder containing the messages you want to display. If the folder name is omitted, MAIL displays a directory of the current folder.

### **QUALIFIERS**

#### **/BEFORE=date**

Lists all the mail messages received before the specified date. Specify the date as DD-MMM-YYYY or specify one of the following keywords: TODAY, TOMORROW, or YESTERDAY. The default is the current date ("today").

#### **/FOLDER**

Lists all the folders contained in the current mail file.

#### **/FULL**

Displays the number of records in the message and indicates whether or not you have replied to the message.

#### **/NEW**

Lists any new (unread) mail messages. When there are no unread messages, MAIL displays the message "No new messages".

#### **/SINCE=date**

Lists all the mail messages received on or after the specified date. Specify the date as DD-MMM-YYYY or specify one of the following keywords: TODAY, TOMORROW, or YESTERDAY. The default is the current date ("today").

**/START=start-point**

Displays a listing of all the mail messages beginning with the message number indicated. If you specify a starting number larger than the number of messages in the current folder, MAIL displays a listing of the last few messages in that folder. To display an alphabetical listing of a group of folders, specify /FOLDER and indicate the first folder name you want to display as *start-point*.

**EXAMPLES**

MAIL> **DIRECTORY/SINCE=11-NOV-1986 DEADLINE**

# From	Date	Subject	DEADLINE
1 BRUTUS::OSGOODE	11-NOV-1986	APPENDIX A	
2 PORTIA::RIPLEY	12-NOV-1986	DEADLINES	
3 UTOPIA::BARBER	3-DEC-1986	A REMINDER	

Lists the sender, date, and subject of mail messages received on or after November 11, 1986, that are currently located in the DEADLINE folder.

MAIL> **DIRECTORY/FOLDER/START=DEMOS**

Listing of folders in WORKDISK: [OSGOODE]MAIL.MAI;1

Press CTRL/C to cancel listing

DEMOS	MAIL
MEMOS	WASTEBASKET

Lists the folders in the file MAIL.MAI;1, beginning with the folder named DEMOS.

**EDIT filename**

By default, the EDIT command invokes the EDT editor, allowing you to edit a file before you send it. If you have copied MAILEDIT.COM from SYS\$SYSTEM and have redefined MAIL\$EDIT to invoke another editor (see Section 1.10.1.3), the EDIT command invokes the other editor. (See Chapter 3 for information about the EDT editor.)

**PARAMETERS**

**filename**

The name of the file you want to edit. If you do not specify a file name, MAIL prompts you for one.

**QUALIFIERS**

**/COMMAND=file-spec (default)**

**/NOCOMMAND**

Indicates the name of a startup command file executed before the editing session begins. The default (for the EDT editor) is the EDTINI.EDT command file.

## **MAIL-12    MAIL              EDIT**

### **/CREATE (default) /NOCREATE**

Determines whether the editor creates a new file when the specified input file is not found. (MAIL prompts you for a file name when you do not specify one on the EDIT command line.)

### **/JOURNAL=file-spec (default) /NOJOURNAL**

Determines whether the editor keeps a journal file during an editing session, and when it does, determines the specification of that journal file. The default (for the EDT editor) is /JOURNAL=filename.JOU, where filename is the name of the file being edited.

### **/OUTPUT=file-spec (default) /NOOUTPUT**

Determines whether the editor creates an output file during the editing session and specifies the name of the output file. The default is /OUTPUT=input-file-spec, in which the version number becomes the highest for that file specification.

### **/READ\_ONLY /NOREAD\_ONLY (default)**

Determines whether both an output file and journal file are created. With the default, /NOREAD\_ONLY, the editor maintains the journal file from which you can recover your edits if an interruption occurs and creates an output file when the EXIT command is entered. Use the /READ\_ONLY qualifier when you are merely reading a file and do not intend to make changes to the file. When you use the /READ\_ONLY qualifier, enter the QUIT command; you must provide a file specification if you want to use the EXIT command to save any changes you have made to the file.

### **/RECOVER /NORECOVER (default)**

Determines whether a journal file is executed before the editing session begins. If the name of the journal file is different from that of the input file, you must specify it with the /JOURNAL qualifier.

### **EXAMPLE**

```
MAIL> EDIT/OUTPUT=STATUS_2.DAT STATUS.DAT
```

Invokes the editor to edit the file STATUS.DAT and names the output file STATUS\_2.DAT.



## **ERASE**

Clears your screen and issues the MAIL> prompt.

## **EXIT**

Returns you to DCL command level. (You may also exit from MAIL by pressing CTRL/Z.)

## **EXTRACT file-spec**

Places a copy of the current message into the specified sequential file.

### **PARAMETERS**

#### **file-spec**

The name of the output file to which the message is copied. The default file type is TXT. By default, the device and the directory will match your current default device and directory.

### **QUALIFIERS**

#### **/ALL**

Copies all the messages in the currently selected folder to the specified output file, separating them with form feeds.

#### **/APPEND**

Adds the current message to the end of the specified file. If the file does not exist, it is created.

#### **/HEADER (default)**

#### **/NOHEADER**

Either includes or removes the header information (From: To: Subject: ) from the mail message.

#### **/MAIL**

Specifies that the output file be a sequential mail file with a default file type of MAI and a protection code of (S:RW,O:RW,G,W). By default, the device and directory will match your current mail file directory. Like /APPEND, /MAIL adds the selected message to the end of the specified file.

## MAIL-14    MAIL EXTRACT

### EXAMPLE

MAIL> 2

#2                    3-APR-1986 10:24:16

MAIL

From:    MEADOW::SMITH  
To:       BRUTUS::OSGOODE  
Subj:    Product Demo

There will be a demonstration of our new product in the Main Lobby  
at 3:30 on Friday. Please try to attend.

MAIL>    EXTRACT/NOHEADER DEMOS.DAT

%MAIL-I-CREATED, WORKDISK:[OSGOODE]DEMOS.DAT;1 created

Places a copy of message number 2 (of the MAIL folder) in the sequential file called DEMOS.DAT. The /NOHEADER qualifier prevents the header information from being copied.

### FILE foldername [filename]

Moves the message most recently read to the specified folder and deletes it from the current folder. Enter CTRL/C to cancel the FILE operation without exiting from MAIL. (FILE is interchangeable with MOVE.)

#### PARAMETERS

##### foldername

The name of the folder to which the current message is moved. If the specified folder does not exist, you are asked whether you want to create it. A folder name can be 1 to 39 characters in length. Valid characters for folder names are A through Z, a through z, \$, —, and 0 through 9.

##### filename

The name of the file to which the current message is moved. If the file name is omitted, the message is moved to the specified folder in the current file.

#### QUALIFIERS

##### /ALL

Moves all the currently selected messages to the specified folder.

##### /CONFIRM (default)

##### /NOCONFIRM

Determines whether or not you will be queried about creating a new folder.

## EXAMPLE

MAIL> 2

#2 3-APR-1986 10:24:16 MAIL  
From: MEADOW::SMITH  
To: BRUTUS::OSGOODE  
Subj: Product Demo

There will be a demonstration of our new product in the Main Lobby at 3:30 on Friday. Please try to attend.

MAIL> FILE DEMOS

Folder DEMOS does not exist.

Do you want to create it (Y/N, default is N)? Y

%MAIL-I-NEWFOLDER, folder DEMOS created

Moves message number 2 from the MAIL folder to the newly created DEMOS folder in the current MAIL file.

## FIRST

Displays the first message in the currently selected folder.

### QUALIFIERS

#### /EDIT

Invokes the editor. You can use the editor to peruse the first message (enter the QUIT command when finished) or you can edit the first message and save the new version in a sequential file (enter the EXIT command and supply a file name).

## FORWARD

Sends a copy of the message most recently read to the specified user or users. MAIL prompts you for the name of the user(s) to whom you want to forward the message. Enter CTRL/C to cancel the FORWARD operation without exiting from MAIL.

### QUALIFIERS

#### /EDIT

Invokes the editor to edit the message you are forwarding.

## MAIL-16    MAIL FORWARD

### /HEADER (default) /NOHEADER

Either includes or removes the header information (From: To: Subj: ) from the message you are forwarding.

#### EXAMPLE

MAIL> 2

#2                      3-APR-1986 10:24:16

MAIL

From: MEADOW::SMITH  
To: BRUTUS::OSGOODE  
Subj: Product Demo

There will be a demonstration of our new product in the Main Lobby at 3:30 on Friday. Please try to attend.

MAIL> FORWARD/NOHEADER

To: BRUTUS::WILSON  
Subj: FYI

Forwards mail message #2, with the original header information removed, to user Wilson on node BRUTUS.

### HELP [topic [subtopic]...]

Displays information about MAIL commands or topics.

#### PARAMETERS

##### topic

The topic about which you want information. To display a list of MAIL topics on which HELP is available, type HELP after the MAIL> prompt.

#### EXAMPLE

MAIL> HELP EXTRACT /APPEND

Displays information about the /APPEND qualifier for the MAIL command EXTRACT.

## **LAST**

Displays the last message in the currently selected folder.

### **QUALIFIERS**

#### **/EDIT**

Invokes the editor. You can use the editor to peruse the last message (enter the QUIT command when finished) or you can edit the last message and save the new version in a sequential file (enter the EXIT command and supply a file name).

## **MAIL [file-spec]**

Sends a message (or file) to another user or users. MAIL prompts you first for the name of the user(s) to whom you are sending the message. You enter the username(s) and/or the file name of the distribution list file(s) in the following format:

**[NODENAME::USERNAME,...][,@LISTNAME]**

Next, MAIL prompts you for the subject of the message. If you include a file specification in the MAIL command, the text of that file is sent to the specified user(s). If you do not specify a file, MAIL prompts you for the text of your message. The message is sent when you press CTRL/Z, or it is canceled when you press CTRL/C. (MAIL is interchangeable with SEND.)

### **PARAMETERS**

#### **file-spec**

The name of the file to be sent.

### **QUALIFIERS**

#### **/EDIT**

#### **/NOEDIT (default)**

Invokes the editor to edit the message you are sending. The EXIT command completes the MAIL operation; the QUIT command cancels the MAIL operation. The /NOEDIT qualifier overrides the MAIL/EDIT default established with the DCL command MAIL/EDIT.

#### **/LAST**

Uses the last message you sent as the text for the message you are currently sending. The /LAST qualifier cannot be used in conjunction with other qualifiers for the MAIL command or a file specification.

## MAIL-18 MAIL MAIL

### **/SELF**

#### **/NOSELF (default)**

Determines whether or not MAIL sends you a copy of the message you are sending. The /NOSELF qualifier overrides the SET COPY\_SELF command.

### **/SUBJECT="subject-text"**

Specifies the subject of the mail message to be sent. The text of the subject must be enclosed in quotation marks.

### EXAMPLE

MAIL> MAIL/SELF

To: BRUTUS::WILSON

Subj: DEADLINES

Enter your message below. Press CTRL/Z when complete or CTRL/C to quit:

What is the deadline on the DAWN project report? CTRL/Z

Sends a message to user WILSON and sends you a copy of that message.

## **MOVE foldername [filename]**

Moves the current message to the specified folder. (MOVE is interchangeable with FILE.)

### PARAMETERS

#### **foldername**

The name of the folder to which the current message is moved. If the specified folder does not exist (and you have not entered the qualifier /NOCONFIRM), you are asked whether you want to create it. A folder name can be 1 to 39 characters in length. Valid characters for folder names are A through Z, a through z, \$, \_, and 0 through 9.

#### **filename**

The name of the file to which the current message is moved. If the specified file does not exist, it is created. If the file name is omitted, the message is moved to the specified folder in the current file.

### QUALIFIERS

#### **/ALL**

Moves all the currently selected messages to the specified folder.

#### **/CONFIRM (default)**

#### **/NOCONFIRM**

Determines whether or not you will be queried about creating a new folder.



## NEXT

Displays the next message.

## QUALIFIERS

### /EDIT

Invokes the editor. You can use the editor to peruse the next message (enter the QUIT command when finished) or you can edit the next message and save the new version in a sequential file (enter the EXIT command and supply a file name).

## PRINT

Queues a copy of the most recently read message for printing. The files created by the PRINT command are not actually released to the print queue until you exit from MAIL so that multiple messages will be concatenated into one print job.

## QUALIFIERS

### /ALL

Queues for printing all messages in the current folder.

### /COPIES=n

Indicates the number of copies to be printed.

### /NOTIFY

Indicates that you will be notified by a broadcast message when the files have been printed.

### /PRINT

Releases all messages previously queued with the PRINT command to the print queue. If you do not specify the /PRINT qualifier, messages are not released to the print queue until you exit from MAIL. The only qualifier you can specify with /PRINT is /NOTIFY.

### /QUEUE=queue-name

Specifies the queue to which a message is to be sent. The default is the last queue name specified (or, if no queue name has been specified, SYS\$PRINT). If you want file separation pages between each MAIL message that you print, send the messages to an execution queue with appropriate default file separation pages. If no queue name is specified, file separation pages are not generated between each MAIL message.

## MAIL-20    MAIL PRINT

### EXAMPLE

MAIL> 2

#2                      3-APR-1986 10:24:16

MAIL

From:    MEADOW::SMITH  
To:       BRUTUS::OSGOODE  
Subj:    Product Demo

There will be a demonstration of our new product in the Main Lobby  
at 3:30 on Friday. Please try to attend.

MAIL> PRINT/QUEUE=LMNO

MAIL> EXIT

Job 434 entered on queue FAST\_PRINT  
Prints message number 2 on printer LMNO.

## PURGE

Deletes all the messages in the WASTEBASKET folder. When you exit from  
MAIL or issue the SET FILE command, an automatic PURGE is done to empty  
the WASTEBASKET folder.

### QUALIFIERS

#### /RECLAIM

Releases deleted message space for reuse. If the /RECLAIM qualifier is omitted,  
an automatic PURGE/RECLAIM is done when the amount of deleted space in a  
MAIL file exceeds the maximum limit.

#### /STATISTICS

Provides a short statistics display indicating the amount of released message  
space when you use the PURGE/RECLAIM command.

### EXAMPLE

MAIL> PURGE/RECLAIM/STATISTICS

Reclaim Statistics:

Data buckets scanned:	14
Data buckets reclaimed:	0
Index buckets reclaimed:	0
Total Buckets reclaimed:	0

Deletes all the messages from the WASTEBASKET folder, releases the deleted  
message space for reuse, and displays information about the deleted message space.

## QUIT

Returns you to DCL command level without deleting any messages that you have previously marked for deletion. Thus, if you accidentally issue the DELETE command, use the QUIT command in place of the EXIT command to exit from MAIL. (The messages you marked for deletion will remain in the WASTEBASKET folder.)

## READ [foldername] [message-number]

Displays a message. READ is the default command for the Mail Utility. Pressing RETURN in response to the MAIL> prompt issues the READ command without parameters. Entering just a number issues the READ command with the message-number parameter. (Pressing RETURN while you are reading a message of several pages displays the next page of the message.) If you issue the READ command after invoking MAIL, MAIL displays the first page of the oldest unread message in your NEWMAIL folder. If there are no unread messages, MAIL displays the oldest message in the MAIL folder. To display a mail message that arrives while you are using the Mail Utility interactively, type:

MAIL> READ/NEW

### PARAMETERS

#### **foldername**

The name of the folder containing the message to be read. If no folder name is specified, MAIL displays a message from the current folder.

#### **message-number**

The number of the message to be read. The message number represents the position of a message in a folder. If you specify a number greater than the number of messages in a folder, MAIL displays the last message in the folder.

### QUALIFIERS

#### **/BEFORE=date**

Displays all the mail messages received before the specified date. Specify the date as DD-MMM-YYYY or specify one of the following keywords: TODAY, TOMORROW, or YESTERDAY. The default is the current date ("today").

#### **/EDIT**

Invokes the editor. You can use the editor to peruse the current message (enter the QUIT command when finished) or you can edit the current message and save the new version in a sequential file (enter the EXIT command and supply a file name).

Enables you to use the editor to edit the message you have received rather than starting a new message.

### **/LAST**

Inserts the message most recently sent as text for the reply. The /LAST qualifier cannot be used with any of the other qualifiers for the REPLY command or with a file specification.

### **/SELF**

#### **/NOSELF (default)**

Determines whether or not MAIL sends you a copy of your response. The default is /NOSELF, unless you have used the SET COPY\_SELF command to specify that copies be sent to you automatically.

### **EXAMPLE**

MAIL> 2

#2

3-APR-1986 10:24:16

MAIL

From: MEADOW::SMITH  
To: BRUTUS::OSGOODE  
Subj: Product Demo

There will be a demonstration of our new product in the Main Lobby at 3:30 on Friday. Please try to attend.

MAIL> REPLY

To: MEADOW::SMITH  
Subj: RE: Product Demo

Enter your message below. Press CTRL/Z when complete or CTRL/C to quit:

Have the programs arrived yet? CTRL/Z

Responds to the sender of message #2.

### **SEARCH [search-string]**

Displays the first message in the current folder that contains the first occurrence of the specified text string. (The entire message is searched, including responses to the From:, To:, and Subj: prompts.)

### **PARAMETERS**

#### **search-string**

The text string that MAIL searches for in the current folder. The search starts with the first message in the current folder. If a search string is not specified, a search is made for the previously specified string, starting after the message you have just read. Uppercase and lowercase differences in the search string are ignored.

## MAIL-24    **MAIL** **SEARCH**

### EXAMPLE

MAIL> **SEARCH** Main Lobby

#2                      3-APR-1986 10:24:16

MAIL

From:    MEADOW::SMITH  
To:       BRUTUS::OSGOODE  
Subj:     Product Demo

There will be a demonstration of our new product in the Main Lobby at 3:30 on Friday. Please try to attend.

Searches for the string "Main Lobby" in the current folder.

### **SELECT** foldername

Establishes a set of messages that you can affect as a group with the following commands:

COPY  
DELETE  
DIRECTORY  
EXTRACT  
MOVE  
READ  
SEARCH

You can also use SELECT to move from one folder to another by specifying a folder name. If you select a folder that does not exist, MAIL displays the following message:

%MAIL-E-NOTEXIST, folder foldername does not exist

### PARAMETERS

#### **foldername**

The name of the folder to be selected. If no folder name is specified, the MAIL folder is selected.

### QUALIFIERS

#### **/BEFORE=date**

Selects mail messages dated before the specified date. Specify the date as DD-MMM-YYYY or specify one of the following keywords: TODAY, TOMORROW, or YESTERDAY. The default is the current date (TODAY).

#### **/NEW**

Selects new (unread) mail messages. When a mail file other than your default mail file is open, MAIL closes that file and opens your default mail file.



**/SINCE=date**

Selects mail messages dated on or after the specified date. Specify the date as DD-MMM-YYYY or specify one of the following keywords: TODAY, TOMORROW, or YESTERDAY. The default is the current date ("today").

**EXAMPLE**

MAIL> **SELECT DEADLINE**

%MAIL-I-SELECTED, 4 messages selected

MAIL> **DIRECTORY**

#	From	Date	Subject	DEADLINE
1	UTOPIA::SABIN	10-SEP-1986	REVIEW DATE	
2	BRUTUS::OSGOODE	11-NOV-1986	APPENDIX A	
3	PORTIA::RIPLEY	12-NOV-1986	DEADLINES	
4	UTOPIA::BARBER	3-DEC-1986	A REMINDER	

Selects the DEADLINE folder and displays its contents.

**SEND [file-spec]**

Sends a message (or file) to another user or users. MAIL prompts you first for the name(s) of the user(s) to whom you are sending the message. You enter the username(s) and/or the file name of a distribution list file in the following format:

[nodename::username,...][,@listname,...]

Next, MAIL prompts you for the subject of the message. If you include a file specification in the SEND command, the text of that file is sent to the specified user(s). If you do not specify a file, MAIL prompts you for the text of your message. The message is sent when you press CTRL/Z, or it is canceled when you press CTRL/C. (SEND is interchangeable with MAIL.)

**PARAMETERS****file-spec**

The name of the file to be sent.

**QUALIFIERS****/EDIT****/NOEDIT (default)**

Invokes the editor to edit the message you are sending. The EXIT command completes the SEND operation; the QUIT command cancels the SEND operation. The /NOEDIT qualifier overrides the SEND/EDIT default established with the DCL command MAIL/EDIT.

## MAIL-26    MAIL SEND

### **/LAST**

Uses the last message you sent as the text for the message you are currently sending. The /LAST qualifier cannot be used with other qualifiers for the SEND command or with a file specification.

### **/SELF**

#### **/NOSELF (default)**

Determines whether or not MAIL sends you a copy of the message you are sending. The /NOSELF qualifier overrides the SET COPY\_SELF command.

### **/SUBJECT="subject-text"**

Specifies the subject of the mail message to be sent. The text of the subject must be enclosed in quotation marks.

#### EXAMPLE

```
MAIL> SEND/EDIT DAWN.DAT
```

```
To:      BRUTUS: :WILSON
```

```
Subj:    PROJECT DAWN
```

```
      .  
      .  
      .
```

```
      CTRL/Z
```

```
*EXIT
```

Sends a file, invoking the editor to edit the file before sending it. The EXIT command completes the SEND operation.

## **SET [NO]AUTO\_PURGE**

Determines whether or not MAIL automatically empties the WASTEBASKET folder when you enter the EXIT or SET FILE commands. The condition you establish with the SET AUTO\_PURGE command remains in effect until you issue the SET NOAUTO\_PURGE command. When you establish the SET NOAUTO\_PURGE condition, you must enter the PURGE command periodically to delete the messages in the WASTEBASKET folder. The default is SET AUTO\_PURGE.

## **SET COPY\_SELF command [,command]**

#### PARAMETERS

#### **command**

One of the following MAIL commands: SEND, REPLY, NOSEND, NOREPLY.

## EXAMPLE

MAIL> SET COPY\_SELF SEND

Enables copies of the mail messages you send to be sent to you.

## SET FILE filename

Opens the file specified by *filename* as the current mail file. Your default mail file is MAIL.MAI. If you have created other mail files with the COPY, FILE, or MOVE commands, you can then use the SET FILE command to open those files. When you enter the SET FILE command, the WASTEBASKET folder of the current mail file is emptied, the file is closed, and the specified (alternate) file is opened.

### PARAMETERS

#### filename

The name of the mail file you are opening.

## EXAMPLE

MAIL> 7

```
#7                      1-APR-1986 12:22:12                      MAIL
From:  EMEER::FRANK
To:    BRUTUS::OSGOODE
Subj:  Dinner at our place
```

My wife and I are having a small group over for dinner this Saturday evening; would you care to join us? R.S.V.P. appreciated.

- Stanley

MAIL> COPY FRANK SOCIAL

%MAIL-S-CREATED, WORKDISK:[OSGOODE]SOCIAL.MAI;1 created  
Folder FRANK does not exist.

Do you want to create it (Y/N, default is N)? Y

%MAIL-I-NEWFOLDER, folder FRANK created

MAIL> SET FILE SOCIAL

MAIL> DIRECTORY FRANK

```
FRANK
# From                      Date                      Subject
1 EMEER::FRANK              1-APR-1986              Dinner at our place
Reads message number 7 in the MAIL folder; uses the COPY command to create a
new file (SOCIAL) and a new folder (FRANK) within that file; opens the mail file
named SOCIAL.MAI with the SET FILE command; and lists the contents of the
folder named FRANK.
```

**MAIL-28    MAIL**  
**SET FOLDER**

**SET FOLDER [foldername]**

Establishes a set of messages that you can affect with the following commands:

COPY  
DELETE  
DIRECTORY  
EXTRACT  
MOVE  
READ  
SEARCH

You can also use the SET FOLDER command to move from one folder to another. (SET FOLDER is interchangeable with SELECT.)

**PARAMETERS**

**foldername**

The name of the folder to be selected. If no folder name is specified, the MAIL folder is selected.

**QUALIFIERS**

**/BEFORE=date**

Selects mail messages dated before the specified date. Specify the date as DD-MMM-YYYY or specify one of the following keywords: TODAY, TOMORROW, or YESTERDAY. The default is the current date ("today").

**/NEW**

Selects new (unread) mail messages. When a mail file other than your default mail file is open, MAIL closes that file and opens your default mail file.

**/SINCE=date**

Selects mail messages dated on or after the specified date. Specify the date as DD-MMM-YYYY or specify one of the following keywords: TODAY, TOMORROW, or YESTERDAY. The default is the current date ("today").

**SET [NO]FORWARD address**

Sets the forwarding address for your mail to the address you specify. The default you establish with the SET FORWARD command remains in effect until you enter the SET NOFORWARD command. The default is SET NOFORWARD.

## PARAMETERS

### address

The address (NODE::USERNAME) to which your mail is forwarded.

## QUALIFIERS

### /USER=username

Indicates the name of another user for whom you are setting a forwarding address. You must have SYSNAM privilege to use the /USER qualifier.

## EXAMPLE

```
MAIL> SET FORWARD MEADOW::SMITH
```

Establishes a forwarding address for user SMITH on node MEADOW.

## SET [NO]MAIL \_DIRECTORY [.subdirectory-name]

Specifies that all your MAI files be moved from your previous mail directory (which by default is SYS\$LOGIN:) to the specified subdirectory. The SET NOMAIL \_DIRECTORY command specifies that all MAI files be moved from the subdirectory back to your mail directory (which by default is SYS\$LOGIN:).

## PARAMETERS

### subdirectory-name

The name of the subdirectory in your SYS\$LOGIN: directory to which all the MAI files are to be moved. The subdirectory name must be preceded by a period.

## QUALIFIERS

### /LOG

Displays a listing of the MAI files moved from the previous directory to the specified subdirectory.

## EXAMPLE

```
$ SHOW TRANSLATION SYS$LOGIN
```

```
SYS$LOGIN = "WORKDISK:[OSGOODE]" (LNM$PROCESS_TABLE)
```

```
$ MAIL
```

```
MAIL> SET MAIL_DIRECTORY [.MESSAGES]
```

```
%MAIL-I-CREATED, WORKDISK:[OSGOODE.MESSAGES] created
```

Enters the DCL command SHOW TRANSLATION SYS\$LOGIN to display the current default directory; issues the DCL command MAIL to enter the interactive Mail Utility; and then creates a subdirectory named OSGOODE.MESSAGES for user Osgoode containing all the MAI files that were previously located in the default directory named OSGOODE.

## MAIL-30    MAIL

### SET [NO]PERSONAL\_NAME

#### SET [NO]PERSONAL\_NAME "text-string"

Enables you to append the specified text string to the end of the "From:" field of mail messages you send. You can fill this field with your full name or any other information. The SET NOPERSONAL\_NAME command clears any name you previously specified with the SET PERSONAL\_NAME command.

#### PARAMETERS

##### text-string

A string of up to 127 characters enclosed in quotation marks. You must begin the string with an alphanumeric character and avoid two consecutive embedded spaces within the string.

#### EXAMPLE

```
MAIL> SET PERSONAL_NAME "Product Design Division"
```

```
MAIL> SEND/SELF
```

```
To:        EMEER::FRANK
```

```
Subj:     R.S.V.P.
```

Enter your message below. Press CTRL/Z when complete or CTRL/C to quit:

I would be happy to join you and Janice for dinner. CTRL/Z

New mail on node BRUTUS from BRUTUS::OSGOODE "Product Design Division"  
Sets the personal name for user OSGOODE to "Product Design Division".

#### SET WASTEBASKET\_NAME foldername

Enables you to change the name of the WASTEBASKET folder, which contains messages that you have selected for deletion. When you change the name of the WASTEBASKET folder while it contains deleted messages, these deleted messages move to the newly named WASTEBASKET folder.

#### PARAMETERS

##### foldername

The name that replaces "WASTEBASKET" for the folder containing deleted messages. The folder name can be any alphanumeric string 1 to 39 characters in length except MAIL or NEWMAIL. Valid characters for folder names are A through Z, a through z, \$, and 0 through 9.

#### SHOW ALL

Displays detailed information about the state of MAIL.



**EXAMPLE**

MAIL> **SHOW ALL**

Your mail file directory is WORKDISK:[OSGOODE].  
Your current mail file is WORKDISK:[OSGOODE.NEWMAIL]MAIL.MAI;1.  
Your current mail folder is MAIL.  
The wastebasket folder name is GARBAGE.  
Mail file WORKDISK:[OSGOODE.NEWMAIL]MAIL.MAI;1  
                 contains 0 deleted message bytes.

You have 3 new messages.

You have not set a forwarding address.  
Your personal name is "John Osgoode, Marketing Representative"  
Automatic copies to yourself are enabled.  
Automatic deleted message purge is disabled.  
Displays information about MAIL for a user named John Osgoode.

**SHOW AUTO\_PURGE**

Displays whether or not MAIL automatically empties the WASTEBASKET folder when you enter the EXIT or FILE command.

**SHOW COPY\_SELF**

Displays whether or not the SEND or REPLY command returns a copy of the message to you.

**SHOW DELETED**

Displays the amount of deleted message space in the current mail file.

**EXAMPLE**

MAIL> **SHOW DELETED**

Mail file WORKDISK:[OSGOODE]MAIL.MAI;1  
                 Contains 2452 deleted message bytes  
Displays the number of deleted message bytes for a user named Osgoode.

**SHOW FILE**

Displays the name of the mail file that is currently open.

## **SHOW FOLDER**

Displays the name of the current folder.

## **SHOW FORWARD**

Displays the names of the specified forwarding addresses.

### **QUALIFIERS**

#### **/USER=username**

Indicates the name of the user whose forwarding address you are displaying. You must have SYSNAM privilege to use the /USER qualifier.

## **SHOW KEY [key-name]**

Displays the key definitions created by the DEFINE/KEY command.

### **PARAMETERS**

#### **key-name**

The name of the key whose definition you want displayed. See the DEFINE /KEY command for a list of the valid key names.

### **QUALIFIERS**

#### **/ALL**

Displays all the key definitions in the specified state or states.

#### **/BRIEF**

Displays only the key definition. (By default, you see all the qualifiers associated with the key definition, including any specified state.)

#### **/DIRECTORY**

Displays the names of all the states for which keys have been defined. If you have not defined any keys, SHOW KEY/DIRECTORY displays the DEFAULT and GOLD states (for the default and GOLD key definitions on the MAIL keypad).

#### **/STATE=(state-name,state-name...)**

Specifies the name of the state for which the specified key definition(s) are to be displayed. If you specify two or more state names, separate them with commas and enclose the list in parentheses.

#### EXAMPLE

MAIL> **SHOW KEY MINUS**

DEFAULT keypad definitions:

MINUS = "READ/NEW" (echo,terminate)

Displays the definition of the minus keypad key. When the minus key was defined, the qualifiers /ECHO (the default) and /TERMINATE were specified.

### **SHOW MAIL\_DIRECTORY**

Displays the name of the device and directory containing all your MAI files.

#### EXAMPLE

MAIL> **SHOW MAIL\_DIRECTORY**

Your mail file directory is WORKDISK:[OSGOODE]

Displays the name of the device and directory containing all the MAI files for user OSGOODE.

### **SHOW NEW\_MAIL\_COUNT**

Displays the number of new (unread) messages.

### **SHOW PERSONAL\_NAME**

Displays the personal name set with the SET PERSONAL\_NAME command.

#### QUALIFIERS

##### **/USER=username**

Shows the personal name for the specified user. You must have SYSNAM privilege to use the /USER qualifier.

### **SHOW WASTEBASKET\_NAME**

Displays the name of the WASTEBASKET folder.

### **SPAWN [command]**

Creates a subprocess of the current process. You can use the SPAWN command to leave MAIL temporarily, perform other functions, and then return to MAIL. The context of the subprocess is copied from the current process.

## **MAIL-34    MAIL              SPAWN**

### **PARAMETERS**

#### **command**

The DCL command string that executes in the context of the created subprocess. When the command completes, the subprocess terminates and control is returned to the parent process. If you do not specify a DCL command, a subprocess is created, transferring control to the DCL command level.

### **QUALIFIERS**

#### **/INPUT=file-spec**

Specifies an input file containing one or more DCL command strings to be executed by the spawned subprocess. If you specify a command string along with an input file, the command string is processed before the commands in the input file. Once processing is complete, the subprocess is terminated.

#### **/LOGICAL \_NAMES (default)**

#### **/NOLOGICAL \_NAMES**

Specifies whether the logical names of the parent process will be copied to the subprocess.

#### **/OUTPUT=file-spec**

Identifies the output file to which the results of the SPAWN operation are to be written. If you omit the /OUTPUT qualifier, output is written to the current SYS\$OUTPUT device. (You should specify an output file other than SYS\$OUTPUT whenever you specify /NOWAIT to prevent output from both processes being displayed simultaneously.)

#### **/PROCESS=subprocess-name**

Specifies the name of the subprocess to be created. The default name of the subprocess is username\_n.

#### **/SYMBOLS (default)**

#### **/NOSYMBOLS**

Determines whether the system passes DCL global and local symbols to the subprocess.

#### **/WAIT (default)**

#### **/NOWAIT**

Controls whether the system waits until the subprocess has completed before allowing more commands to be specified. The /NOWAIT qualifier allows you to specify new commands while the specified subprocess is running. (You should use the /OUTPUT qualifier with /NOWAIT to prevent output from both processes being displayed simultaneously.)

## EXAMPLE

```
MAIL> SPAWN/NOWAIT/OUTPUT=LOG.DAT DIFFERENCES OLD.DAT NEW.DAT
```

Spawns a subprocess to enter the DCL command DIFFERENCES, writing the output from the DIFFERENCES command to the file LOG.DAT. The /NOWAIT qualifier enables you to enter other commands while the subprocess is running.

## MAIL.3 MAIL Keypad

The MAIL keypad allows you to enter MAIL commands by pressing the appropriate keypad key. The command you enter appears on your terminal screen after the MAIL> prompt.

### MAIL.3.1 MAIL Keypad Diagram

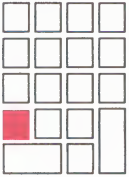
By default, the keypad keys on the VT200 and VT100 series terminals are defined to execute the following MAIL commands:

Hex Values	0	1	2	3	4	5	6	7
0	NUL	DLE	SP	0	@	P	`	p
1	SOH	DC1	!	1	A	Q	a	q
2	STX	DC2	"	2	B	R	b	r
3	ETX	DC3	#	3	C	S	c	s
4	EOT	DC4	\$	4	D	T	d	t
5	ENQ	NAK	%	5	E	U	e	u
6	ACK	SYN	&	6	F	V	f	v
7	BEL	ETB	'	7	G	W	g	w
8	BS	CAN	(	8	H	X	h	x
9	HT	EM	)	9	I	Y	i	y
A	LF	SUB	*	:	J	Z	j	z
B	VT	ESC	+	;	K	[	k	{
C	FF	FS	,	<	K	\	l	
D	CR	GS	-	=	M	]	m	}
E	SO	RS	.	>	N	^	n	~
F	SI	US	/	?	O	_	o	DEL

### **MAIL.3.2 MAIL Keypad Commands**

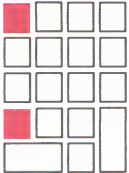
The diagram above the command description represents the keypad shown in Section MAIL.3.1. MAIL keypad commands execute immediately when you press the keypad key, except for those commands for which the /NOTERMINATE qualifier is specified.

#### **BACK**



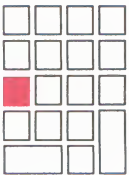
Displays the previous message.

#### **BACK/EDIT**



Invokes the editor and displays the previous message.

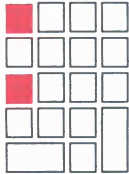
#### **CURRENT**



Displays the beginning of the message you are currently reading.

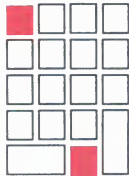


**CURRENT/EDIT**



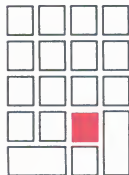
Invokes the editor and displays the beginning the message you are currently reading.

**DELETE/NOTERMINATE**



Enters the DELETE command but does not execute it until you press the RETURN key. (Entering CTRL/C will cancel the DELETE operation and keep you within MAIL.)

**DIRECTORY**



Displays a list of the messages contained in the current folder, listing message number, sender's name and node, date, and subject.

**DIRECTORY MAIL**



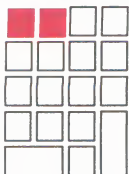
Displays the contents of the MAIL folder.

**MAIL-38**

**MAIL**

**MAIL Keypad**

**DIRECTORY/FOLDER**



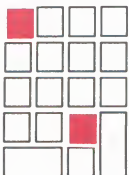
Lists all the folders contained in the current mail file.

**DIRECTORY/NEW**



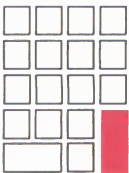
Lists any new (unread) messages.

**DIRECTORY/START=99999**



Displays a listing of the messages in the current folder beginning with message number 99,999. This command usually lists the last messages in the folder because you seldom have 99,999 messages in a folder.

**ENTER**



Enters MAIL commands. (ENTER is interchangeable with RETURN.)

**ERASE/NOECHO**

Clears your screen and issues the MAIL> prompt.

## EXTRACT

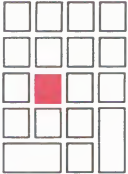
Places a copy of the current message into the specified sequential file. (MAIL prompts you for the name of the file.) You must press RETURN to execute the EXTRACT command.

**EXTRACT/MAIL**

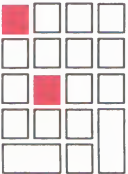
Places a copy of the current message at the end of the specified mail file. (MAIL prompts you for the name of the file, which has a default file type of MAI.) You must press RETURN to execute the EXTRACT/MAIL command.

## FILE

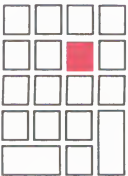

Moves the current message to the specified folder. (MAIL prompts you for the folder name.) You must press RETURN to execute the FILE command.

**FIRST**

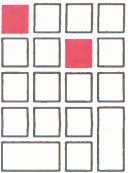
Displays the first message in the currently selected folder.

**FIRST/EDIT**

Invokes the editor and displays the first message in the current folder.

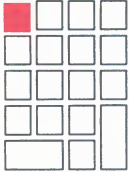
**FORWARD**

Sends a copy of the most recently read message to the specified users. MAIL prompts you for the names of the users to whom you want to forward the message.

**FORWARD/EDIT**

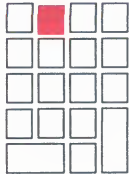
Invokes the editor to edit the most recently read message before forwarding it to the specified users. The EXIT command completes the FORWARD operation; the QUIT command cancels the FORWARD operation.

**GOLD**



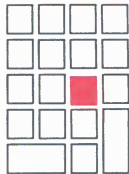
When pressed before another keypad key, specifies that key's alternate function. GOLD can be used to redefine keypad keys and certain keyboard keys (See the DEFINE/KEY command in Section MAIL.2.1).

**HELP**



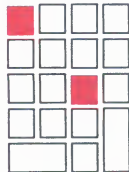
Displays a list of all the mail commands and topics on which HELP is available.

**LAST**



Displays the last message in the currently selected folder.

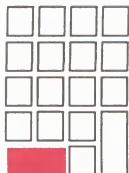
**LAST/EDIT**



Invokes the editor and displays the last message in the currently selected folder.

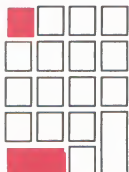
**MAIL-42    MAIL**  
**MAIL Keypad**

**NEXT**



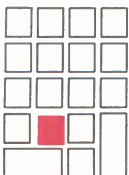
Displays the next message in the currently selected folder.

**NEXT/EDIT**



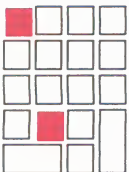
Invokes the editor and displays the next message in the currently selected folder.

**PRINT**



Queues a copy of the most recently read message for printing. The files created by the PRINT command are not actually released to the print queue until you exit from MAIL, so that multiple messages will be concatenated into one print job.

**PRINT/PRINT/NOTIFY**



Immediately releases all messages previously queued to the print queue with the PRINT command and notifies you by a broadcast message when the files have been printed.

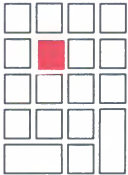


**READ/NEW**



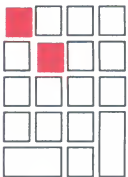
Displays new messages received while you are in MAIL. If there are no new messages, the message "No new messages" will be displayed.

**REPLY**



Enters the REPLY command and then prompts you for the text of your reply. (CTRL/C cancels the reply operation and returns you to the MAIL> prompt.)

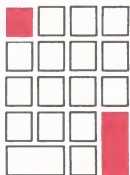
**REPLY/EDIT**



Enters the REPLY command and then invokes the editor for you to enter the text of your reply. The EXIT command completes the REPLY operation; the QUIT command cancels the REPLY operation.

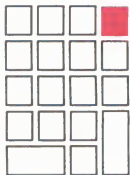
**MAIL-44      MAIL**  
**MAIL Keypad**

**SELECT/NOTERMINATE**



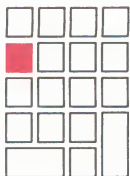
Enters the **SELECT** command so that you may then enter the name of the folder containing the messages you want to select. You must press **RETURN** to execute the **SELECT** command.

**SELECT MAIL**



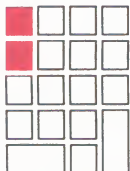
Selects and moves to the **MAIL** folder.

**SEND**



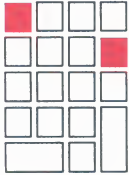
Enters the **SEND** command and then prompts you for the name of the user to whom you are sending the message, the subject of the message, and the text of the message. The message is sent when you press **CTRL/Z**, or it is cancelled when press **CTRL/C**.

**SEND/EDIT**



Enters the SEND command, prompting you for the name of the user to whom you are sending the message and the subject of the message, and then invokes the editor for you to enter your mesesage. The EXIT command sends the message; the QUIT command cancels the message.

**SHOW NEW\_MAIL\_COUNT**



Displays the number of new (unread) messages.



# Index

## A

### Abbreviation

DSR command, 4-1, DSR-10

### /ABORT

DISMOUNT, DCL-54

REPLY, DCL-107

STOP/QUEUE, DCL-201

Absolute time, 5-14

ACCEPT flag, DSR-46

### Access

object, 7-4

### Access control entry, 7-8

ALARM\_JOURNAL, 7-10, ACL-1

creating, 7-8

DEFAULT\_PROTECTION, 7-10, ACL-1

deleting, 7-16

format, ACL-1

IDENTIFIER, 7-8, ACL-1

processing, 7-20

replacing, 7-16

### Access control list, 7-7, ACL-1

copying, 7-16

default protection, 7-11

EDIT/ACL command, 7-15

editing, 7-16

identifier, 7-7

modifying, 7-16

SET ACL command, 7-15

SET DEVICE/ACL command, 7-15

SET DIRECTORY/ACL command, 7-15

SET FILE/ACL command, 7-15

SHOW ACL command, 7-14

### Access control list editor

See ACL Editor

### /ACCESSED

INITIALIZE, DCL-63

### /ACCESSED (cont'd.)

MOUNT, DCL-87

SET VOLUME, DCL-165

### /ACCESS MODE

SHOW LOGICAL, DCL-172

### Access mode

logical name, 2-30

### /ACCOUNT

ACCOUNTING, DCL-1

### Account, 7-1

turnkey, 1-2

### /ACCOUNTING

RUN, DCL-112

SHOW PROCESS, DCL-176

### ACCOUNTING command, DCL-1

### ACE

See Access control entry

### /ACL

DIRECTORY, DCL-50

SET ACL, ACL-16

### ACL editor

cursor direction, ACL-7

deleting text, 7-19, ACL-8

EDIT/ACL command, 7-17

finding text, ACL-9

GOLD key, ACL-10

invoking, ACL-4

journal file, ACL-5

keypad, 7-17, ACL-6

keypad commands, ACL-7

moving the cursor, 7-19

recovering, ACL-5

restoring text, 7-20, ACL-12

### /ADDRESS

ACCOUNTING, DCL-2

Address sort, 2-39

## Index-2

### /ADJUST

SET WORKING\_SET, DCL-167

ADVANCE command (ACL editor  
command), ACL-7

ADVANCE command (EDT), 3-14, EDT-4

### /ADVANCED\_VIDEO

SET TERMINAL, DCL-156

ADV FIELD command (ACL editor  
command), ACL-7

### /AFTER

PRINT, DCL-95

SET ACL, ACL-16

SET DEVICE/ACL, ACL-18

SET DIRECTORY/ACL, ACL-20

SET FILE/ACL, ACL-22

SET QUEUE/ENTRY, DCL-148

SUBMIT, DCL-203

.AJ (DSR command), DSR-11

### /ALARM

SET AUDIT, DCL-121

ALARM\_JOURNAL access control entry,  
7-10

ACCESS field, 7-10

### /ALIGN

START/QUEUE, DCL-193

### /ALL

DEALLOCATE, DCL-32

DEASSIGN, DCL-32

DELETE/KEY, DCL-43

DELETE/SYMBOL, DCL-44

RECALL, DCL-104

REPLY, DCL-107

SHOW KEY, DCL-171

SHOW LOGICAL, DCL-172

SHOW MEMORY, DCL-174

SHOW PROCESS, DCL-176

SHOW QUEUE, DCL-177

SHOW SYMBOL, DCL-180

Allocate access category, 7-4

ALLOCATE command, 2-21, DCL-8

### /ALLOCATED

SHOW DEVICES, DCL-169

### /ALLOCATION

APPEND, DCL-9

COPY, DCL-25

MERGE, DCL-79

### /ALLOCATION (cont'd.)

SORT, DCL-183

Alphabetic lists

DSR, 4-12

### /ALTYPEAHD

SET TERMINAL, DCL-156

### /ANSI\_CRT

SET TERMINAL, DCL-156

ANSWER command (MAIL), MAIL-3

.AP (DSR command), DSR-11

### /APPEND

OPEN, DCL-94

APPEND command, DCL-9

APPEND command (EDT), EDT-4

.APPENDIX (DSR command), 4-16, DSR-11

### /APPLICATION\_KEYPAD

SET TERMINAL, DCL-156

Arrow keys (EDT), EDT-14

### /ASCII

DEPOSIT, DCL-45

EXAMINE, DCL-59

### ASCII

collating sequence, 2-40

US (default) character set, ESC-3

ASCII Character Set, CHAR-1

ASSIGN/MERGE command, DCL-14

ASSIGN/QUEUE command, DCL-14

ASSIGN command, 2-21, DCL-12

Assignment statement, 5-17

### /ASSIST

MOUNT, DCL-87

.AST (DSR command), DSR-12

### /AST\_LIMIT

RUN, DCL-112

.AT (DSR command), DSR-12

ATTACH command, 1-41, DCL-14

ATTACH command (MAIL), MAIL-4

### /ATTRIBUTES

CREATE/NAME\_TABLE, DCL-30

### /AUTHORIZE

RUN, DCL-112

Authorize Utility, 7-1

### /AUTOBAUD

SET TERMINAL, DCL-156

.AUTOJUSTIFY (DSR command), DSR-11



## /AUTOMATIC

MOUNT, DCL-87  
 Automatic login, 1-2  
 .AUTOPARAGRAPH (DSR command),  
     DSR-11  
 .AUTOSUBTITLE (DSR command), DSR-12  
 .AUTOTABLE (DSR command), DSR-12  
 /AVAILABLE  
     SET DEVICE, DCL-126  
 .AX (DSR command), DSR-11

**B**

.B (DSR command), DSR-12  
 BACK/EDIT keypad command (MAIL),  
     MAIL-36  
 BACK command (MAIL), MAIL-4  
 BACK keypad command (MAIL), MAIL-36  
 /BACKSPACE  
     RUNOFF, DSR-1  
 BACKSPACE (F12), ACL-14  
 BACKSPACE key (F12), EDT-15  
 /BACKUP  
     APPEND, DCL-9  
     BACKUP, DCL-15  
     COPY, DCL-25  
     DELETE, DCL-41  
     DIRECTORY, DCL-50  
     PRINT, DCL-95  
     PURGE, DCL-101  
     RENAME, DCL-105  
     SET DIRECTORY, DCL-127  
     SET FILE, DCL-129  
     SUBMIT, DCL-204  
     TYPE, DCL-208  
 BACKUP command, DCL-15  
 BACKUP command (ACL editor command),  
     ACL-7  
 BACKUP command (EDT), 3-15, EDT-5  
 /BACKWARD  
     START/QUEUE, DCL-193  
 /BADBLOCKS  
     INITIALIZE, DCL-63  
 Balance set, 1-36, 1-38

## /BASE\_PRIORITY

INITIALIZE/QUEUE, DCL-68  
 SET QUEUE, DCL-143  
 START/QUEUE, DCL-193

## /BATCH

INITIALIZE/QUEUE, DCL-68  
 SHOW QUEUE, DCL-177  
 SHOW SYSTEM, DCL-180  
 START/QUEUE, DCL-193

## Batch job, 1-43

controlling, 1-45  
 deleting, 1-45  
 execution time, 1-45  
 job number, 1-44  
 log file, 1-46  
 naming, 1-45  
 output, 1-46  
 parameters, 1-44  
 restarting, 1-45  
 submitting, 1-44

## Baud rate, 1-14, 1-17

.BB (DSR command), DSR-12

## /BEFORE

ACCOUNTING, DCL-2  
 APPEND, DCL-9  
 BACKUP, DCL-16  
 COPY, DCL-25  
 DELETE, DCL-41  
 DIRECTORY, DCL-50  
 LIBRARY, DCL-74  
 PRINT, DCL-95  
 PURGE, DCL-101  
 RENAME, DCL-105  
 SET ACL, ACL-16  
 SET DIRECTORY, DCL-127  
 SET DIRECTORY/ACL, ACL-20  
 SET FILE, DCL-129  
 SET FILE/ACL, ACL-22  
 SUBMIT, DCL-204  
 TYPE, DCL-209

.BEGIN BAR (DSR command), DSR-12

## /BELL

REPLY, DCL-107

## /BINARY

ACCOUNTING, DCL-2

Binary data, 5-1

## Index-4

- /BIND
  - MOUNT, DCL-87
- Bit, 5-1
- .BLANK (DSR command), DSR-12
- /BLOCK\_COUNT
  - SET RMS\_DEFAULT, DCL-154
- /BLOCK\_LIMIT
  - INITIALIZE/QUEUE, DCL-68
  - SET QUEUE, DCL-143
  - START/QUEUE, DCL-193
- /BLOCK\_MODE
  - SET TERMINAL, DCL-156
- /BLOCK\_SIZE
  - BACKUP, DCL-16
- /BLOCKS
  - DUMP, DCL-55
- BLOCKS
  - LIBRARY option, DCL-74
- /BLOCKSIZE
  - MOUNT, DCL-87
- /BOLD
  - RUNOFF, DSR-2
  - RUNOFF/CONTENTS, DSR-7
- BOLD flag, DSR-46
- Bolding text
  - DSR, 4-14
- BOTTOM command (ACL editor command),  
ACL-7
- BOTTOM command (EDT), 3-14, EDT-5
- .BR (DSR command), DSR-13
- /BRDCSTMBX
  - SET TERMINAL, DCL-156
- .BREAK (DSR command), DSR-13
- BREAK flag, DSR-46
- /BRIEF
  - ACCOUNTING, DCL-2
  - BACKUP, DCL-16
  - DIRECTORY, DCL-50
  - LOGOUT, DCL-78
  - SHOW DEVICES, DCL-169
  - SHOW KEY, DCL-171
  - SHOW QUEUE, DCL-177
  - SHOW QUEUE/FORM, DCL-178
- /BROADCAST
  - SET TERMINAL, DCL-157

- /BUCKET\_SIZE
  - MERGE, DCL-79
  - SORT, DCL-183
- /BUFFER\_COUNT
  - BACKUP, DCL-16
  - SET RMS\_DEFAULT, DCL-154
- /BUFFER\_LIMIT
  - RUN, DCL-112
- Buffering
  - terminal, 1-8
- Bulleted list
  - DSR, 4-11
- /BURST
  - PRINT, DCL-95
  - SET QUEUE/ENTRY, DCL-148
- /BY\_OWNER
  - APPEND, DCL-9
  - COPY, DCL-25
  - DELETE, DCL-41
  - DIRECTORY, DCL-50
  - PRINT, DCL-95
  - PURGE, DCL-101
  - RENAME, DCL-105
  - SET ACL, ACL-16
  - SET DIRECTORY, DCL-127
  - SET DIRECTORY/ACL, ACL-20
  - SET FILE, DCL-129
  - SET FILE/ACL, ACL-22
  - SUBMIT, DCL-204
  - TYPE, DCL-209
- /BYTE
  - DEPOSIT, DCL-45
  - DUMP, DCL-55
  - EXAMINE, DCL-59
- Byte, 5-1

## C

- .C (DSR command), DSR-13
- /CACHE
  - MOUNT, DCL-88
- CALL command, 6-24, DCL-22
- CANCEL command, DCL-23
- CAPITALIZE flag, DSR-46

- /CARRIAGE\_CONTROL
  - SET PROMPT, DCL-141
- CARRIAGE\_CONTROL
  - SPAWN, DCL-191
- Carriage control, 2-13
- .CC (DSR command), DSR-14
- .CENTER (DSR command), DSR-13
- Center text
  - DSR, 4-6
- .CH (DSR command), DSR-13
- /CHANGE\_BAR
  - DIFFERENCES, DCL-46
- /CHANGE\_BARS
  - RUNOFF, DSR-2
- CHANGE command (EDT), 3-8, EDT-22
- .CHAPTER (DSR command), 4-16, DSR-13
- Character data, 5-1, 5-3
  - alphanumeric, 5-3
  - expression, 5-7
  - literal, 5-5
  - name, 5-4
  - nonprintable, 5-3
  - special, 5-3
- /CHARACTERISTIC
  - INITIALIZE/QUEUE, DCL-68
  - START/QUEUE, DCL-193
- /CHARACTERISTICS
  - PRINT, DCL-96
  - SET QUEUE, DCL-143
  - SET QUEUE/ENTRY, DCL-148
  - SUBMIT, DCL-204
- Character set
  - ASCII, CHAR-1
  - DEC Multinational, CHAR-5
- CHAR command (EDT), 3-10, 3-21, EDT-5
- /CHECK\_SEQUENCE
  - MERGE, DCL-79
- CHNGCASE command (EDT), EDT-5
- CLEAR command (EDT), EDT-23
- /CLI
  - SET QUEUE/ENTRY, DCL-148
  - SPAWN, DCL-191
  - SUBMIT, DCL-204
- CLOSE command, 2-22, 6-11, DCL-23
- CLUSTER\_SIZE
  - INITIALIZE, DCL-63
- Collating sequence
  - ASCII, 2-40
  - EBCDIC, 2-40
  - multinational, 2-40
- /COLUMNS
  - DIRECTORY, DCL-50
- /COMMAND
  - EDIT, EDT-1
  - EDIT/TPU, DCL-57
- Command
  - See Also Command procedure
  - abbreviating, 1-20
  - capitalization, 1-19
  - continuation over many lines, 1-19
  - delimiter, 1-19
  - foreign, 1-33
  - format, 1-19
  - interactive, 1-18, 1-23
  - interrupting, 1-23
  - parameter, 1-20
  - tailoring, 1-27
- COMMAND command (EDT), 3-8, EDT-6
- Command file
  - EDT, 3-32, EDT-1
  - VAXTPU, DCL-57
- Command image, 1-33
- Command level, 1-2
  - nesting, 6-3
- Command line
  - recalling, 1-32
- Command-line editing, 1-24
- Command procedure, 1-28, 6-1
  - batch, 1-43
  - case statement, 6-20
  - cleanup, 6-30
  - coding, 6-16
  - conditional command, 6-16
  - conditional statement, 6-19
  - CTRL/Y, 6-30
  - data line, 6-4
  - debugging, 6-26
  - error handling, 6-28
  - executing, 6-2
  - exiting, 6-3
  - file I/O, 6-10
  - format, 6-2

## Index-6

### Command procedure (cont'd.)

- global symbol, 6-8
- I/O errors, 6-16
- input, 6-4
- inputting from file, 6-7
- inputting from terminal, 6-7
- logic, 6-16
- loops, 6-21
- output, 6-9
- output to terminal, 6-9
- parameter, 6-4
- passing a literal, 6-4
- passing a symbol, 6-5
- passing data, 6-4
- passing parameters, 6-5
- returning data, 6-8
- return status, 6-3
- SET DEFAULT command, 1-44
- stubs, 6-25
- subroutines, 6-23
- testing and debugging, 6-25
- using comments, 6-2
- variables, 6-16
- writing, 6-16

### Command qualifier, 1-21

#### Commands

- DSR, DSR-10

#### /COMMENT

- BACKUP, DCL-16

- MOUNT, DCL-88

- .COMMENT (DSR command), DSR-10, DSR-14

#### /COMMENT\_DELIMITER

- DIFFERENCES, DCL-46

- COMMENT flag, DSR-46

#### /COMPARE

- BACKUP, DCL-16

#### /COMPRESS

- LIBRARY, DCL-74

- COMPRESS command (MAIL), MAIL-5

#### /CONCATENATE

- COPY, DCL-25

- Concealed device name, 2-24

#### /CONFIRM

- APPEND, DCL-10

- BACKUP, DCL-16

#### /CONFIRM (cont'd.)

- COPY, DCL-25

- DELETE, DCL-41

- PRINT, DCL-96

- RENAME, DCL-105

- SET ACL, ACL-16

- SET DIRECTORY, DCL-127

- SET DIRECTORY/ACL, ACL-20

- SET FILE, DCL-129

- SET FILE/ACL, ACL-22

- SET PROTECTION, DCL-142

- SUBMIT, DCL-204

- TYPE, DCL-209

- UNLOCK, DCL-210

- /CONFIRM/PURGE, DCL-101

- CONNECT command, DCL-24

#### /CONTENTS

- RUNOFF, DSR-7

#### /CONTIGUOUS

- APPEND, DCL-10

- COPY, DCL-26

- MERGE, DCL-80

- SORT, DCL-183

#### /CONTINUE

- CONNECT, DCL-24

- DISCONNECT, DCL-53

- CONTINUE command, 1-24, 3-9, DCL-24

#### /CONTINUOUS

- SHOW PROCESS, DCL-176

#### Control characters, 1-7

- ASCII values, KEY-2

- CTRL/A, ACL-15, EDT-17

- CTRL/B, 1-27, 1-32

- CTRL/C, 1-23, 1-24, 1-50, 3-6, EDT-17

- CTRL/D, ACL-15, EDT-17

- CTRL/E, EDT-17

- CTRL/H, ACL-15, EDT-18

- CTRL/I, EDT-18

- CTRL/J, ACL-15, EDT-18

- CTRL/K, 3-32, EDT-18, EDT-20

- CTRL/L, EDT-20

- CTRL/M, EDT-20

- CTRL/O, 1-8

- CTRL/Q, 1-8

- CTRL/R, EDT-20

- CTRL/S, 1-8

## Control characters (cont'd.)

CTRL/T, 1-23, EDT-20  
 CTRL/U, 3-16, 3-18, ACL-15, EDT-20  
 CTRL/V, 1-7  
 CTRL/W, 3-9, ACL-15, EDT-21  
 CTRL/Y, 1-6, 1-23, 3-9, 6-30  
 CTRL/Z, 1-50, 2-14, 3-8, ACL-15,  
 EDT-21

.CONTROL CHARACTERS (DSR  
 command), DSR-14

CONTROL flag, DSR-46  
 period, DSR-10

## /COPIES

PRINT, DCL-96  
 SET QUEUE/ENTRY, DCL-148

COPY command, 2-14, 2-53, DCL-24  
 DSR output, 4-23

COPY command (EDT), EDT-23

COPY command (MAIL), 1-57, MAIL-6

## /CPUDEFAULT

INITIALIZE/QUEUE, DCL-69  
 SET QUEUE, DCL-144  
 START/QUEUE, DCL-194

## /CPUMAXIMUM

INITIALIZE/QUEUE, DCL-69  
 SET QUEUE, DCL-144  
 START/QUEUE, DCL-194

## /CPUTIME

SET QUEUE/ENTRY, DCL-148  
 SUBMIT, DCL-205

## /CR

SET PRINTER, DCL-137

## /CRC

BACKUP, DCL-17

## /CREATE

EDIT, EDT-1  
 EDIT/TPU, DCL-57  
 LIBRARY, DCL-75

CREATE/DIRECTORY command, 2-10,  
 7-12, DCL-29

CREATE/NAME\_TABLE command, 2-29,  
 DCL-30

CREATE command, 2-14, DCL-28

## /CREATED

APPEND, DCL-10  
 BACKUP, DCL-17

## /CREATED (cont'd.)

COPY, DCL-26  
 DELETE, DCL-41  
 DIRECTORY, DCL-50  
 PRINT, DCL-96  
 PURGE, DCL-101  
 RENAME, DCL-105  
 SET ACL, ACL-17  
 SET DIRECTORY, DCL-128  
 SET DIRECTORY/ACL, ACL-20  
 SET FILE, DCL-130  
 SET FILE/ACL, ACL-23  
 SUBMIT, DCL-205  
 TYPE, DCL-209

Creating a scrolling region, ESC-7

## /CRFILL

SET TERMINAL, DCL-157

## /CROSS\_REFERENCE

LIBRARY, DCL-75

CTRL/A, ACL-15, EDT-17

CTRL/B, 1-27, 1-32

CTRL/C, 1-23, 1-24, 1-50, 3-6, EDT-17

CTRL/D, ACL-15, EDT-17

CTRL/E, EDT-17

CTRL/H, ACL-15, EDT-18

CTRL/I, EDT-18

CTRL/J, ACL-15, EDT-18

CTRL/K, 3-32, EDT-18, EDT-20

CTRL/L, EDT-20

CTRL/M, EDT-20

CTRL/O, 1-8

CTRL/Q, 1-8

CTRL/R, EDT-20

CTRL/S, 1-8

CTRL/T, 1-23, EDT-20

CTRL/U, 3-16, 3-18, ACL-15, EDT-20

CTRL/V, 1-7

CTRL/W, 3-9, ACL-15, EDT-21

CTRL/Y, 1-6, 1-23, 3-9, 6-30

command procedure, 6-28

CTRL/Z, 1-50, 3-8, ACL-15, EDT-21

CURRENT/EDIT keypad command (MAIL),  
 MAIL-37

CURRENT command (MAIL), MAIL-7

CURRENT keypad command (MAIL),  
 MAIL-36

CUT command (EDT), 3-24, EDT-6

## D

.D (DSR command), DSR-14

/DATA

LIBRARY, DCL-75

Data

passing to command procedure, 6-4

/DATA\_CHECK

INITIALIZE, DCL-63

MOUNT, DCL-88

SET FILE, DCL-130

SET VOLUME, DCL-165

Data representation, 5-1

binary, 5-1

character, 5-1, 5-3, 5-4, 5-7

expression, 5-7

logical, 5-11

logical data, 5-6

numeric, 5-5, 5-9

symbol, 5-16

Data storage, 5-1

Data type, 2-12

/DATE

DIRECTORY, DCL-50

Date

system base, LEX-3

.DATE (DSR command), DSR-14

.DAX (DSR command), DSR-15

.DBB (DSR command), DSR-14

.DBO (DSR command), DSR-14

.DCH (DSR command), DSR-16

DCL command

RUN, DCL-111

DCL command level, 1-2

DCL commands

ACCOUNTING, DCL-1

ALLOCATE, 2-21, DCL-8

APPEND, DCL-9

ASSIGN, 2-21, DCL-12

ASSIGN/MERGE, DCL-14

ASSIGN/QUEUE, DCL-14

ATTACH, 1-41, DCL-14

BACKUP, DCL-15

CALL, 6-24, DCL-22

DCL commands (cont'd.)

CANCEL, DCL-23

CLOSE, 2-22, 6-11, DCL-23

CONNECT, DCL-24

CONTINUE, 1-24, 3-9, DCL-24

COPY, 2-14, 2-53, DCL-24

CREATE, 2-14, DCL-28

CREATE/DIRECTORY, 2-10, 7-12,  
DCL-29

CREATE/NAME\_TABLE, 2-29, DCL-30

DEALLOCATE, DCL-31

DEASSIGN, 2-22, DCL-32

DEASSIGN/QUEUE, DCL-33

DECK, 6-4, DCL-33

DEFINE, 2-21, DCL-34

DEFINE/CHARACTERISTIC, DCL-36

DEFINE/FORM, DCL-36

DEFINE/KEY, 1-29, DCL-38

DELETE, 2-17, 2-50, DCL-40

DELETE/CHARACTERISTIC, DCL-42

DELETE/ENTRY, DCL-42

DELETE/FORM, DCL-43

DELETE/INTRUSION\_RECORD,  
DCL-43

DELETE/KEY, 1-31, DCL-43

DELETE/QUEUE, DCL-44

DELETE/SYMBOL, 5-17, DCL-44

DEPOSIT, DCL-45

DIFFERENCES, DCL-46

DIRECTORY, 2-9, 7-14, DCL-49

DISCONNECT, DCL-53

DISMOUNT, 2-22, DCL-54

DUMP, 5-4, DCL-54

EDIT, 1-24, 2-17, 3-2, EDT-1

EDIT/ACL, 7-16, ACL-4

EOD, 6-4, DCL-58

EXAMINE, DCL-59

EXIT, 6-3, DCL-60

GOSUB, 6-23, DCL-60

GOTO, 6-19, DCL-60

HELP, 1-26, DCL-60

IF, 6-19, DCL-62

INITIALIZE, DCL-62

INITIALIZE/QUEUE, DCL-68

INQUIRE, 6-6, DCL-73



## DCL commands (cont'd.)

LIBRARY, 2-46, DCL-74  
 LOGOUT, 1-6, 1-41, DCL-78  
 MAIL, 1-49, 2-53, MAIL-1  
 MERGE, 2-45, DCL-79  
 MOUNT, 2-21, 7-13, DCL-86  
 ON, 6-28, DCL-93  
 ON CONTROL\_Y, 6-30  
 OPEN, 2-15, 2-21, 6-10, DCL-94  
 PRINT, 2-33, DCL-94  
 PURGE, 2-18, DCL-100  
 READ, 2-15, 6-7, 6-12, DCL-102  
 RECALL, 1-27, 1-32, DCL-104  
 RENAME, 2-14, DCL-104  
 REPLY, DCL-106  
 REQUEST, DCL-110  
 RETURN, 6-23, DCL-111  
 RUN, 1-33, 1-34, DCL-111  
 RUNOFF, 4-19  
 RUNOFF/CONTENTS, DSR-1, DSR-7  
 RUNOFF/INDEX, DSR-1  
 SEARCH, DCL-117  
 SET ACCOUNTING, DCL-119  
 SET ACL, 7-15, 7-16, ACL-15  
 SET AUDIT, DCL-120  
 SET BROADCAST, DCL-124  
 SET CONTROL, DCL-125  
 SET CONTROL=Y, 6-30  
 SET DAY, DCL-125  
 SET DEFAULT, 2-4, DCL-126  
 SET DEVICE, DCL-126  
 SET DEVICE/ACL, 7-15, ACL-18  
 SET DIRECTORY, DCL-127  
 SET DIRECTORY/ACL, 7-15, ACL-19  
 SET FILE, DCL-129  
 SET FILE/ACL, 7-15, ACL-22  
 SET HOST, 1-5, DCL-132  
 SET KEY, DCL-132  
 SET LOGINS/INTERACTIVE, DCL-133  
 SET MAGTAPE, DCL-133  
 SET MESSAGE, 1-49, DCL-134  
 SET ON, 6-29, DCL-135  
 SET OUTPUT\_RATE, DCL-135  
 SET PASSWORD, 1-1, DCL-136  
 SET PRINTER, DCL-136  
 SET PROCESS, DCL-138

## DCL commands (cont'd.)

SET PROMPT, DCL-141  
 SET PROTECTION, 7-11, 7-12,  
     DCL-141  
 SET PROTECTION/DEFAULT,  
     DCL-142  
 SET PROTECTION/DEVICE, 7-4,  
     DCL-142  
 SET QUEUE, DCL-143  
 SET QUEUE/ENTRY, DCL-147  
 SET RESTART\_VALUE, DCL-152  
 SET RIGHTS\_LIST, DCL-153  
 SET RMS\_DEFAULT, DCL-154  
 SET SYMBOL, 5-18, DCL-155  
 SET TERMINAL, 1-7, 1-8, 1-41,  
     DCL-155  
 SET TIME, DCL-163  
 SET UIC, 7-3, DCL-164  
 SET VERIFY, 6-26, DCL-164  
 SET VOLUME, DCL-164  
 SET WORKING\_SET, DCL-167  
 SHOW ACCOUNTING, DCL-167  
 SHOW ACL, 7-14  
 SHOW AUDIT, DCL-168  
 SHOW BROADCAST, DCL-168  
 SHOW DEFAULT, DCL-168  
 SHOW DEVICES, 7-14, DCL-168  
 SHOW ERROR, DCL-170  
 SHOW INTRUSION, DCL-170  
 SHOW KEY, 1-31, DCL-171  
 SHOW LOGICAL, 2-22, DCL-172  
 SHOW MAGTAPE, DCL-173  
 SHOW MEMORY, 1-36, DCL-174  
 SHOW NETWORK, 1-3, DCL-175  
 SHOW PRINTER, DCL-175  
 SHOW PROCESS, 1-35, 7-14, DCL-175  
 SHOW PROTECTION, 7-14, DCL-177  
 SHOW QUEUE, 2-32, DCL-177  
 SHOW QUEUE/CHARACTERISTICS,  
     DCL-178  
 SHOW QUEUE/FORM, DCL-178  
 SHOW QUOTA, DCL-178  
 SHOW RMS\_DEFAULT, DCL-179  
 SHOW STATUS, 1-36, DCL-179  
 SHOW SYMBOL, 6-27, DCL-179  
 SHOW SYSTEM, 1-38, DCL-180

## Index-10

### DCL commands (cont'd.)

- SHOW TERMINAL, 1-7, DCL-181
- SHOW TIME, DCL-181
- SHOW TRANSLATION, DCL-181
- SHOW USERS, DCL-182
- SHOW WORKING\_SET, DCL-182
- SORT, 2-38, DCL-182
- SPAWN, 1-39, DCL-190
- START/QUEUE, DCL-192
- START/QUEUE/MANAGER, DCL-199
- STOP, 1-24, 6-3, 6-30, DCL-200
- STOP/ABORT, DCL-202
- STOP/ENTRY, DCL-202
- STOP/QUEUE, 2-34, DCL-200
- STOP/QUEUE/MANAGER, DCL-202
- STOP/REQUEUE, DCL-202
- SUBMIT, 1-44, DCL-203
- SYNCHRONIZE, DCL-208
- TYPE, 2-17, 6-10, DCL-208
- UNLOCK, DCL-210
- WAIT, DCL-210
- WRITE, 2-15, 6-9, 6-11, DCL-211

DEALLOCATE command, DCL-31

DEASSIGN/QUEUE command, DCL-33

DEASSIGN command, 2-22, DCL-32

/DEBUG

- RUN, DCL-111
- RUNOFF, DSR-2

/DEC\_CRT

- SET TERMINAL, DCL-157

/DECIMAL

- DEPOSIT, DCL-45
- DUMP, DCL-55
- EXAMINE, DCL-59

DECK command, 6-4, DCL-33

DEC Multinational Character Set, CHAR-2

DECnet

- See Also Network, Dynamic asynchronous DECnet, and Static asynchronous DECnet

DECnet-VAX

- access, 2-54
- file manipulation, 2-52
- logical name, 2-52
- MAIL, 2-53
- node, 2-53

### DECnet-VAX (cont'd.)

- printing files, 2-53
- /DEEPEST\_LEVEL
- RUNOFF/CONTENTS, DSR-7
- /DEFAULT
- INITIALIZE/QUEUE, DCL-69
- SET ACL, ACL-17
- SET DAY, DCL-125
- SET DIRECTORY/ACL, ACL-21
- SET FILE/ACL, ACL-23
- SET QUEUE, DCL-144
- START/QUEUE, DCL-194

Default

- device name designation, 2-3

DEFAULT\_PROTECTION access control entry, 7-10

Default protection, 7-5, 7-11

DEFINE/CHARACTERISTIC, DCL-36

DEFINE/FORM, DCL-36

DEFINE/KEY command, 1-29, DCL-38

DEFINE/KEY command (MAIL), MAIL-7

DEFINE command, 2-21, DCL-34

DEFINE KEY command (EDT), 3-32, 3-35, EDT-24

DEFINE MACRO command (EDT), 3-36, EDT-27

Defining keys, 1-29

- examining key definitions, 1-31
- list of definable keys, 1-30

DEL ACE command (ACL editor command), ACL-8

/DELAY

- RUN, DCL-112

DEL C command (ACL editor command), ACL-8

DEL C command (EDT), 3-17, EDT-6

DEL EOL command (ACL editor command), ACL-8

DEL EOL command (EDT), 3-18, EDT-7

/DELETE

- BACKUP, DCL-17
- LIBRARY, DCL-76
- PRINT, DCL-96
- READ, DCL-103
- SET ACL, ACL-17
- SET DEVICE/ACL, ACL-19

- /DELETE (cont'd.)
  - SET DIRECTORY/ACL, ACL-21
  - SET FILE/ACL, ACL-23
  - SET MESSAGE, DCL-135
  - SUBMIT, DCL-205
- DELETE ( <X> ), ACL-14
- DELETE/CHARACTERISTIC command, DCL-42
- DELETE/ENTRY command, DCL-42
- DELETE/FORM command, DCL-43
- DELETE/INTRUSION\_RECORD command, DCL-43
- DELETE/KEY command, DCL-43
- DELETE/NOTERMINATE keypad command (MAIL), MAIL-37
- DELETE/QUEUE command, DCL-44
- DELETE/SYMBOL command, 5-17, DCL-44
- Delete access category, 7-4
- DELETE command, 2-17, 2-50, DCL-40
- DELETE command (EDT), EDT-28
- DELETE command (MAIL), MAIL-9
- DELETE key ( <X> ), EDT-15
- DELETE KEY command, 1-31
- Deleting
  - access control entry, 7-16
  - directory, 2-10
  - file, 2-17
  - key definition, 1-31
  - libraries, 2-50
  - library modules, 2-51
  - logical name definitions, 2-22
  - logical name table, 2-29
  - mail file, 1-58
  - mail folder, 1-57
  - mail message, 1-55
  - password, 1-1
  - queue entry, 1-22
  - system logical names, 2-25
- DEL L command (EDT), 3-18, EDT-7
- Delta time, 5-15
- DEL W command (ACL editor command), ACL-9
- DEL W command (EDT), 3-17, EDT-7
- /DENSITY
  - BACKUP, DCL-17
  - DENSITY (cont'd.)
    - INITIALIZE, DCL-63
    - MOUNT, DCL-89
    - SET MAGTAPE, DCL-133
  - DEPOSIT command, DCL-45
  - /DESCENDANTS
    - SHOW LOGICAL, DCL-172
  - /DESCRIPTION
    - DEFINE/FORM, DCL-37
  - /DETACHED
    - RUN, DCL-112
  - Detached process, 1-34, 7-3
  - /DEVICE
    - PRINT, DCL-97
    - RUNOFF, DSR-2
    - SHOW QUEUE, DCL-177
  - Device, 2-1
    - concealed name, 2-24
    - default, 2-4
    - default name designation, 2-3
    - generic name, 2-2
    - mass-storage, 2-1
    - name, 2-2
    - protection, 7-13
    - record-oriented, 2-1
    - spooling, 2-34
    - type, 2-2
  - /DEVICE\_TYPE
    - SET TERMINAL, DCL-157
  - .DHL (DSR command), DSR-17
  - .DHY (DSR command), DSR-14
  - /DIAL
    - SET HOST, DCL-132
  - Dialing in, 1-2
  - /DIALUP
    - SET TERMINAL, DCL-157
  - DIFFERENCES command, DCL-46
  - DIGITAL Standard Runoff, 4-1, DSR-1
  - Digital Standard Runoff
    - commands, DSR-10
    - flags, DSR-45
  - /DIRECTORIES
    - INITIALIZE, DCL-64
  - /DIRECTORY
    - SHOW KEY, DCL-172

## Index-12

### Directory

- creating, 2-10
- default, 2-4
- deleting, 2-10
- displaying, 2-9
- first-level, 2-8, 2-10
- hierarchy, 2-3, 2-9
- master, 2-8, 2-19
- protection, 7-12
- rooted, 2-20
- subdirectory, 2-10
- system, 2-19
- top-level, 2-3

DIRECTORY/FOLDER keypad command  
(MAIL), MAIL-37

DIRECTORY/NEW keypad command  
(MAIL), MAIL-38

DIRECTORY/START=99999 keypad  
command (MAIL), MAIL-38

DIRECTORY command, 2-9, 7-14, DCL-49

DIRECTORY command (MAIL), 1-58,  
MAIL-10

DIRECTORY keypad command (MAIL),  
MAIL-37

DIRECTORY MAIL keypad command  
(MAIL), MAIL-37

/DISABLE

- REPLY, DCL-107, DCL-108
- SET ACCOUNTING, DCL-120
- SET AUDIT, DCL-121

/DISABLE\_SWAPPING

- INITIALIZE/QUEUE, DCL-70
- SET QUEUE, DCL-145
- START/QUEUE, DCL-195

.DISABLE BAR (DSR command), DSR-14

.DISABLE BOLDING (DSR command),  
DSR-14

.DISABLE HYPHENATION (DSR  
command), DSR-14

.DISABLE INDEXING (DSR command),  
DSR-15

.DISABLE OVERSTRIKING (DSR  
command), DSR-15

.DISABLE TOC (DSR command), DSR-15

.DISABLE UNDERLINING (DSR  
command), DSR-15

### /DISCONNECT

- SET TERMINAL, DCL-158

DISCONNECT command, DCL-53

/DISK

- SET RMS\_DEFAULT, DCL-154
- SHOW QUOTA, DCL-179

### Disk

- protection, 7-13

### Diskette

- See Also Disk

### Disk file

- See File

### Disk save set

- See Save set

### /DISMISS

- SET TERMINAL, DCL-158

DISMOUNT command, 2-22, DCL-54

### /DISPLAY

- EDIT/TPU, DCL-57

### Display

- library, 2-49

.DISPLAY APPENDIX (DSR command),  
DSR-15

.DISPLAY CHAPTER (DSR command),  
DSR-16

.DISPLAY ELEMENTS (DSR command),  
DSR-16

.DISPLAY LEVELS (DSR command),  
DSR-17

.DISPLAY NUMBER (DSR command),  
DSR-17

.DISPLAY SUBPAGE (DSR command),  
DSR-18

### Distribution list

- in MAIL, 1-52

.DIX (DSR command), DSR-15

.DLE (DSR command), DSR-16

.DNM (DSR command), DSR-17

DO command (EDT), EDT-16

### Document layout

- DSR, 4-15

### /DOLLARS

- DECK, DCL-34

.DOV (DSR command), DSR-15

### /DOWN

- RUNOFF, DSR-3

.DSP (DSR command), DSR-18

## DSR

See DIGITAL Standard Runoff  
flag, 4-4

## DSR commands

.APPENDIX (.AX), DSR-11  
.AUTOJUSTIFY (.AJ), DSR-11  
.AUTOPARAGRAPH (.AP), DSR-11  
.AUTOSUBTITLE (.AST), DSR-12  
.AUTOTABLE (.AT), DSR-12  
.BEGIN BAR (.BB), DSR-12  
.BLANK (.B), DSR-12  
.BREAK (.BR), DSR-13  
.CENTER (.C), DSR-13  
.CHAPTER (.CH), DSR-13  
.COMMENT (!), DSR-14  
.CONTROL CHARACTERS (.CC),  
DSR-14  
.DATE (.D), DSR-14  
.DISABLE BAR (.DBB), DSR-14  
.DISABLE BOLDING (.DB0), DSR-14  
.DISABLE HYPHENATION (.DHY),  
DSR-14  
.DISABLE INDEXING (.DIX), DSR-15  
.DISABLE OVERSTRIKING (.DOV),  
DSR-15  
.DISABLE TOC (.DTC), DSR-15  
.DISABLE UNDERLINING (.DUL),  
DSR-15  
.DISPLAY APPENDIX (.DAX), DSR-15  
.DISPLAY CHAPTER (.DCH), DSR-16  
.DISPLAY ELEMENTS (.DLE), DSR-16  
.DISPLAY LEVELS (.DHL), DSR-17  
.DISPLAY NUMBER (.DNM), DSR-17  
.DISPLAY SUBPAGE (.DSP), DSR-18  
.ELSE, DSR-18  
.ENABLE BAR (.EBB), DSR-18  
.ENABLE BOLDING (.EBO), DSR-18  
.ENABLE HYPHENATION (.EHY),  
DSR-18  
.ENABLE INDEXING (.EIX), DSR-19  
.ENABLE OVERSTRIKING (.EOV),  
DSR-19  
.ENABLE TOC (.ETC), DSR-19  
.ENABLE UNDERLINING (.EUN),  
DSR-19

## DSR commands (cont'd.)

.END BAR (.EB), DSR-19  
.END FOOTNOTE (.EFN), DSR-19  
.ENDIF, DSR-20  
.END LIST (.ELS), DSR-20  
.END LITERAL (.EL), DSR-20  
.END NOTE (.EN), DSR-20  
.END SUBPAGE (.ES), DSR-20  
.ENTRY (Y), DSR-20  
.FIGURE (.FG), DSR-21  
.FIGURE DEFERRED (.FGD), DSR-21  
.FILL (.F), DSR-21  
.FIRST TITLE (.FT), DSR-21  
.FLAGS ACCEPT (.FL ACCEPT), DSR-21  
.FLAGS ALL (.FL), DSR-22  
.FLAGS BOLD (.FL BOLD), DSR-22  
.FLAGS BREAK (.FL BREAK), DSR-22  
.FLAGS CAPITALIZE (.FL CAPITALIZE),  
DSR-22  
.FLAGS COMMENT (.FL COMMENT),  
DSR-22  
.FLAGS CONTROL (.FL CONTROL),  
DSR-22  
.FLAGS HYPHENATE (.FL  
HYPHENATE), DSR-23  
.FLAGS INDEX (.FL INDEX), DSR-23  
.FLAGS LOWERCASE (.FL  
LOWERCASE), DSR-23  
.FLAGS OVERSTRIKE (.FL  
OVERSTRIKE), DSR-23  
.FLAGS PERIOD (.FL PERIOD), DSR-23  
.FLAGS SPACE (.FL SPACE), DSR-23  
.FLAGS SUBINDEX (.FL SUBINDEX),  
DSR-24  
.FLAGS SUBSTITUTE (.FL  
SUBSTITUTE), DSR-24  
.FLAGS UNDERLINE (.FL UNDERLINE),  
DSR-24  
.FLAGS UPPERCASE (.FL UPPERCASE),  
DSR-24  
.FOOTNOTE (.FN), DSR-24  
.HEADER LEVEL (.HL), DSR-25  
.HEADERS [ON] (.HD), DSR-26  
.HEADERS LOWER (.HD LOWER),  
DSR-25



## Index-14

### DSR commands (cont'd.)

.HEADERS MIXED (.HD MIXED),  
DSR-26  
.HEADERS UPPER (.HD UPPER),  
DSR-26  
.IF, DSR-26  
.IFNOT, DSR-26  
.INDENT (.I), DSR-27  
.INDEX (.X), DSR-27  
.JUSTIFY (.J), DSR-27  
.KEEP (.K), DSR-27  
.LAYOUT (.LO), DSR-28  
.LEFT MARGIN (.LM), DSR-28  
.LIST (.LS), DSR-29  
.LIST ELEMENT (.LE), DSR-29  
.LITERAL (.LT), DSR-30  
.NO AUTOJUSTIFY (.NAJ), DSR-30  
.NO AUTOPARAGRAPH (.NAP),  
DSR-30  
.NO AUTOSUBTITLE (.NAST), DSR-30  
.NO AUTOTABLE (.NAT), DSR-30  
.NO CONTROL CHARACTERS (.NCC),  
DSR-30  
.NO DATE (.ND), DSR-30  
.NO FILL (.NF), DSR-31  
.NO FLAGS [ALL] (.NFL [ALL]), DSR-31  
.NO FLAGS ACCEPT (.NFL ACCEPT),  
DSR-31  
.NO FLAGS BOLD (.NFL BOLD),  
DSR-31  
.NO FLAGS BREAK (.NFL BREAK),  
DSR-31  
.NO FLAGS CAPITALIZE (.NFL  
CAPITALIZE), DSR-31  
.NO FLAGS COMMENT (.NFL  
COMMENT), DSR-31  
.NO FLAGS CONTROL (.NFL  
CONTROL), DSR-31  
.NO FLAGS HYPHENATE (.NFL  
HYPHENATE), DSR-32  
.NO FLAGS INDEX (.NFL INDEX),  
DSR-32  
.NO FLAGS LOWERCASE (.NFL  
LOWERCASE), DSR-32  
.NO FLAGS OVERSTRIKE (.NFL  
OVERSTRIKE), DSR-32

### DSR commands (cont'd.)

.NO FLAGS PERIOD (.NFL PERIOD),  
DSR-32  
.NO FLAGS SPACE (.NFL SPACE),  
DSR-32  
.NO FLAGS SUBINDEX (.NFL  
SUBINDEX), DSR-32  
.NO FLAGS SUBSTITUTE (.NFL  
SUBSTITUTE), DSR-32  
.NO FLAGS UNDERLINE (.NFL  
UNDERLINE), DSR-33  
.NO FLAGS UPPERCASE (.NFL  
UPPERCASE), DSR-33  
.NO JUSTIFY (.NJ), DSR-33  
.NO KEEP (.NK), DSR-33  
.NO NUMBER (.NNM), DSR-33  
.NO PAGING (.NPA), DSR-33  
.NO PERIOD (.NPR), DSR-33  
.NO SPACE (.NSP), DSR-33  
.NO SUBTITLE (.NST), DSR-34  
.NOTE (.NT), DSR-34  
.NUMBER [PAGE] (.NMPG), DSR-36  
.NUMBER APPENDIX (.NMAX),  
DSR-34  
.NUMBER CHAPTER (.NMCH), DSR-35  
.NUMBER LEVEL (.NMLV), DSR-35  
.NUMBER LIST (.NMLS), DSR-36  
.NUMBER RUNNING (.NMR), DSR-36  
.NUMBER SUBPAGE (.NMSPG),  
DSR-37  
.PAGE (.PG), DSR-37  
.PAGE SIZE (.PS), DSR-37  
.PAGING (.PA), DSR-38  
.PARAGRAPH (.P), DSR-38  
.PERIOD (.PR), DSR-39  
.REPEAT (.RPT), DSR-39  
.REQUIRE (.REQ), DSR-39  
.RESTORE (.RE), DSR-39  
.RIGHT (.R), DSR-39  
.RIGHT MARGIN (.RM), DSR-40  
RUNOFF, DSR-1  
.SAVE (.SA), DSR-40  
.SEND TOC (.STC), DSR-40  
.SET DATE (.SDT), DSR-40  
.SET LEVEL (.SL), DSR-41  
.SET PARAGRAPH (.SPR), DSR-41



## DSR commands (cont'd.)

.SET TIME (.STM), DSR-42  
 .SKIP (.S), DSR-42  
 .SPACING (.SP), DSR-43  
 .STYLE HEADERS (.STHL), DSR-43  
 .SUBPAGE (.SPG), DSR-44  
 .SUBTITLE (.ST), DSR-44  
 .TAB STOPS (.TS), DSR-44  
 .TEST PAGE (.TP), DSR-44  
 .TITLE (.T), DSR-45  
 .VARIABLE (.VR), DSR-45  
 .XLOWER (.XL), DSR-45  
 .XUPPER (.XU), DSR-45  
 .DTC (DSR command), DSR-15  
 /DTE  
   SET HOST, DCL-132  
 /DUAL\_PORT  
   SET DEVICE, DCL-126  
 .DUL (DSR command), DSR-15  
 /DUMP  
   RUN, DCL-113  
   SET PROCESS, DCL-139  
 DUMP command, 5-4, DCL-54  
 /DUPLICATES  
   MERGE, DCL-80  
   SORT, DCL-183

**E**

.EB (DSR command), DSR-19  
 .EBB (DSR command), DSR-18  
 .EBO (DSR command), DSR-18  
 /ECHO  
   DEFINE/KEY, DCL-39  
   SET TERMINAL, DCL-158  
 EDIT  
   /READ\_ONLY, 2-17  
 /EDIT\_MODE  
   SET TERMINAL, DCL-158  
 EDIT/ACL command, ACL-4  
   CTRL/H, ACL-15  
   CTRL/J, ACL-15  
   CTRL/U, ACL-15  
   CTRL/Z, ACL-15  
   GOLD + CTRL/Z, ACL-15

## EDIT/ACL keypad commands

ADVANCE, ACL-7  
 ADV FIELD, ACL-7  
 BACKUP, ACL-7  
 BOTTOM, ACL-7  
 DEL ACE, ACL-8  
 DEL C, ACL-8  
 DEL EOL, ACL-8  
 DEL W, ACL-9  
 ENTER, ACL-9  
 EOL, ACL-9  
 FIELD, ACL-9  
 FIND, ACL-9  
 FNDNXT, ACL-10  
 GOLD, ACL-10  
 HELP, ACL-11  
 HELP FMT, ACL-11  
 INSERT, ACL-11  
 ITEM, ACL-11  
 MOVE SCREEN, ACL-11  
 OVER ACE, ACL-12  
 TOP, ACL-12  
 UND ACE, ACL-12  
 UND C, ACL-12  
 UND W, ACL-13  
 WORD, ACL-13

EDIT command, 2-17, 3-2, EDT-1

EDIT command (MAIL), MAIL-11

## Editing

  command line, 1-24

## EDT

  nonprintable characters, 5-4

## EDT commands

  SHOW CASE, EDT-43

## EDT commands

  ADVANCE, 3-14, EDT-4  
   APPEND, EDT-4  
   BACKUP, 3-15, EDT-5  
   BOTTOM, 3-14, EDT-5  
   CHANGE, 3-8, EDT-22  
   CHAR, 3-10, 3-21, EDT-5  
   CHNGCASE, EDT-5  
   CLEAR, EDT-23  
   COMMAND, 3-8, EDT-6  
   COPY, EDT-23  
   CTRL/A, EDT-17

## Index-16

### EDT commands (cont'd.)

CTRL/C, EDT-17  
CTRL/D, EDT-17  
CTRL/H, EDT-18  
CTRL/I, EDT-18  
CTRL/J, EDT-18  
CTRL/K, EDT-18, EDT-20  
CTRL/L, EDT-20  
CTRL/M, EDT-20  
CTRL/R, EDT-20  
CTRL/T, EDT-20  
CTRL/W, EDT-21  
CTRL/Z, 3-8, EDT-21  
CUT, 3-24, EDT-6  
DEFINE KEY, 3-32, 3-35, EDT-24  
DEFINE MACRO, 3-36, EDT-27  
DEL C, 3-17, EDT-6  
DEL EOL, 3-18, EDT-7  
DELETE, EDT-28  
DEL L, 3-18, EDT-7  
DEL W, 3-17, EDT-7  
DO, EDT-16  
ENTER, 3-8, EDT-7  
EOL, 3-11, EDT-8  
EXIT, 3-2, EDT-29  
FILL, 3-27, 3-28, EDT-8, EDT-30  
FIND, 3-20, EDT-8, EDT-16, EDT-30  
FNDNXT, 3-22, EDT-8  
GOLD, 3-5, EDT-9  
HELP, 3-7, EDT-9, EDT-16, EDT-31  
INCLUDE, 3-29, EDT-31  
INSERT, EDT-32  
INSERT HERE, EDT-16  
LINE, 3-12, EDT-9  
MOVE, EDT-32  
NEXT, EDT-50  
NEXT SCREEN, EDT-17  
OPEN LINE, 3-12, EDT-10  
PAGE, 3-13, EDT-10  
PASTE, 3-24, EDT-11  
PREV SCREEN, EDT-17  
PRINT, EDT-33  
QUIT, 3-3, EDT-33  
REMOVE, EDT-17  
REPLACE, EDT-11, EDT-34  
RESEQUENCE, EDT-34

### EDT commands (cont'd.)

RESET, 3-24, EDT-11  
SECT, 3-13, EDT-12  
SELECT, 3-19, EDT-12, EDT-17  
SET AUTOREPEAT, EDT-34  
SET CASE, EDT-35  
SET COMMAND, EDT-35  
SET CURSOR, EDT-36  
SET ENTITY, 3-34, EDT-36  
SET FNF, EDT-37  
SET HELP, EDT-37  
SET KEYPAD, EDT-37  
SET LINES, 3-33, EDT-38  
SET MODE, 3-34, EDT-38  
SET NUMBERS, 3-19, EDT-38  
SET QUIET, 3-34, EDT-38  
SET REPEAT, EDT-39  
SET SCREEN, 3-33, 3-34, EDT-39  
SET SEARCH, 3-20, 3-24, EDT-39  
SET SUMMARY, EDT-40  
SET TAB, EDT-40, EDT-48  
SET TERMINAL, EDT-41  
SET TEXT, EDT-41  
SET TRUNCATE, 3-33, EDT-42  
SET VERIFY, EDT-42  
SET WORD DELIMITER, EDT-42  
SET WRAP, 3-27, 3-34, EDT-42  
SHOW AUTOREPEAT, EDT-43  
SHOW BUFFER, 3-31, EDT-43  
SHOW COMMAND, EDT-44  
SHOW CURSOR, EDT-44  
SHOW ENTITY, EDT-44  
SHOW FILES, EDT-44  
SHOW FNF, EDT-45  
SHOW HELP, EDT-45  
SHOW KEY, EDT-45  
SHOW KEYPAD, EDT-45  
SHOW LINES, EDT-46  
SHOW MODE, EDT-46  
SHOW NUMBERS, EDT-46  
SHOW QUIET, EDT-46  
SHOW REPEAT, EDT-47  
SHOW SCREEN, EDT-47  
SHOW SEARCH, EDT-47  
SHOW SUMMARY, EDT-47  
SHOW TERMINAL, EDT-48

## EDT commands (cont'd.)

SHOW TEXT, EDT-48  
 SHOW TRUNCATE, EDT-48  
 SHOW VERIFY, EDT-48  
 SHOW VERSION, EDT-49  
 SHOW WORD, EDT-49  
 SHOW WRAP, EDT-49  
 SPECINS, EDT-12  
 SUBS, EDT-13  
 SUBSTITUTE, 3-22, EDT-49  
 TAB ADJUST, EDT-51  
 TOP, 3-13, EDT-13  
 TYPE, EDT-52  
 UND C, 3-17, EDT-13  
 UND L, 3-18, EDT-14  
 UND W, 3-17, EDT-14  
 WORD, 3-11, EDT-14  
 WRAP, 3-27  
 WRITE, 3-29, EDT-52

## EDT editor

buffers, 3-29, EDT-30, EDT-43  
 command file, 3-32, EDT-1  
 copying text, EDT-23  
 create, EDT-1  
 cursor direction, 3-10, 3-14, 3-15, EDT-4,  
 EDT-5  
 cutting text, 3-24, EDT-4  
 defining keys, 3-35, EDT-18, EDT-24  
 defining macros, 3-36, EDT-27  
 deleting text, 3-16, EDT-6  
 finding text, 3-20, EDT-8  
 function keys, EDT-26  
 GOLD key, EDT-9, EDT-26  
 inserting text, 3-16, EDT-32  
 invoking EDT, 3-2, EDT-1  
 journal file, 3-9, EDT-2  
 keypad, 3-4, EDT-3  
 keypad commands, 3-4, EDT-3  
 line-editing commands, 3-3, 3-8,  
 EDT-21, EDT-22  
 moving text, 3-24, EDT-32  
 nokeypad commands, EDT-53  
 pasting text, 3-24, EDT-11  
 range specification, EDT-12, EDT-21  
 reading from a file, 3-29, EDT-31  
 recovering edits, 3-9, EDT-2

## EDT editor (cont'd.)

replacing text, 3-22, EDT-11, EDT-13,  
 EDT-34, EDT-49  
 screen format, 3-33, EDT-39  
 /RECOVER qualifier, 3-9, EDT-2  
 start-up file, 3-32, EDT-1  
 supplemental keypad, 3-5  
 terminating EDT, 3-2, EDT-29  
 writing to a file, 3-29, EDT-52  
 .EFN (DSR command), DSR-19  
 .EHY (DSR command), DSR-18  
 /EIGHT\_BIT  
 SET TERMINAL, DCL-158  
 .EIX (DSR command), DSR-19  
 .EL (DSR command), DSR-20  
 .ELS (DSR command), DSR-20  
 .ELSE (DSR command), DSR-18  
 .EN (DSR command), DSR-20  
 /ENABLE  
 SET ACCOUNTING, DCL-120  
 SET AUDIT, DCL-121  
 /ENABLE\_GENERIC  
 INITIALIZE/QUEUE, DCL-70  
 SET QUEUE, DCL-145  
 START/QUEUE, DCL-195  
 .ENABLE BAR (DSR command), DSR-18  
 .ENABLE BOLDING (DSR command),  
 DSR-18  
 .ENABLE HYPHENATION (DSR  
 command), DSR-18  
 .ENABLE INDEXING (DSR command),  
 DSR-19  
 .ENABLE OVERSTRIKING (DSR  
 command), DSR-19  
 .ENABLE TOC (DSR command), DSR-19  
 .ENABLE UNDERLINING (DSR command),  
 DSR-19  
 /END\_OF\_FILE  
 READ, DCL-103  
 SET FILE, DCL-130  
 SET MAGTAPE, DCL-134  
 .END BAR (DSR command), DSR-19  
 .END FOOTNOTE (DSR command),  
 DSR-19  
 .ENDIF (DSR command), DSR-20  
 .END LIST (DSR command), DSR-20

## Index-18

.END LITERAL (DSR command), DSR-20  
.END NOTE (DSR command), DSR-20  
.END SUBPAGE (DSR command), DSR-20  
ENDSUBROUTINE command, 6-24  
/ENQUEUE\_LIMIT  
    RUN, DCL-113  
/ENTENSION  
    MOUNT, DCL-89  
/ENTER  
    SET FILE, DCL-130  
ENTER command (ACL editor command),  
    ACL-9  
ENTER command (EDT), 3-8, EDT-7  
ENTER keypad command (MAIL), MAIL-38  
/ENTRY  
    ACCOUNTING, DCL-2  
    SET QUEUE/ENTRY, DCL-148  
    STOP QUEUE, DCL-201  
    SYNCHRONIZE, DCL-208  
.ENTRY (DSR command), DSR-20  
EOB symbol (EDT), 3-2  
EOD command, 6-4, DCL-58  
EOL command (ACL editor command),  
    ACL-9  
EOL command (EDT), 3-11, EDT-8  
.EOV (DSR command), DSR-19  
Equivalence name, 2-21  
/ERASE  
    DEFINE/KEY, DCL-39  
    DELETE, DCL-42  
    INITIALIZE, DCL-64  
    PURGE, DCL-101  
/ERASE\_ON\_DELETE  
    SET FILE, DCL-130  
    SET VOLUME, DCL-165  
ERASE/NOECHO keypad command  
    (MAIL), MAIL-39  
ERASE command (MAIL), MAIL-13  
/ERROR  
    CLOSE, DCL-23  
    READ, DCL-103  
    RUN, DCL-113  
    WRITE, DCL-211  
/ERROR\_LOGGING  
    SET DEVICE, DCL-126

/ERRORB  
    OPEN, DCL-94  
Error condition, 1-48  
    \$SEVERITY, 1-48  
    \$STATUS, 1-48  
Error message  
    format, 1-49  
.ES (DSR command), DSR-20  
/ESCAPE  
    SET TERMINAL, DCL-158  
Escape sequence, 1-8, ESC-1  
    creating scrolling region, ESC-7  
    erasing screen display, ESC-6  
    moving the cursor, ESC-2  
    specifying character set, ESC-3  
    specifying display characteristic, ESC-6  
    using in command procedure, 1-8  
    using in LOCAL mode, 1-9  
    using in text file, 1-9  
ESC key, 1-8  
.ETC (DSR command), DSR-19  
.EUN (DSR command), DSR-19  
/EXACT  
    SEARCH, DCL-118  
EXAMINE command, DCL-59  
/EXCLUDE, DCL-130  
    APPEND, DCL-10  
    BACKUP, DCL-17  
    COPY, DCL-26  
    DELETE, DCL-42  
    DIRECTORY, DCL-51  
    PRINT, DCL-97  
    PURGE, DCL-102  
    RENAME, DCL-106  
    SEARCH, DCL-118  
    SET ACL, ACL-17  
    SET DIRECTORY, DCL-128  
    SET DIRECTORY/ACL, ACL-21  
    SET FILE/ACL, ACL-23  
    SUBMIT, DCL-205  
    TYPE, DCL-209  
Executable image  
    See Program  
Execute access category, 7-4  
Execute procedure, 6-24  
Execute procedure (@), 6-2

- /EXECUTIVE\_MODE
  - ASSIGN, DCL-12
  - CREATE/NAME\_TABLE, DCL-30
  - DEASSIGN, DCL-32
  - DEFINE, DCL-34
- EXIT command, 1-24, 6-3, DCL-60
- EXIT command (EDT), 3-2, EDT-29
- EXIT command (MAIL), MAIL-13
- /EXPIRATION\_DATE
  - SET FILE, DCL-130
- /EXPIRED
  - APPEND, DCL-10
  - BACKUP, DCL-17
  - COPY, DCL-26
  - DELETE, DCL-42
  - DIRECTORY, DCL-51
  - PRINT, DCL-97
  - PURGE, DCL-102
  - RENAME, DCL-106
  - SET DIRECTORY, DCL-128
  - SUBMIT, DCL-205
  - TYPE, DCL-209
- Expression
  - character, 5-7
  - logical, 5-11
  - numeric, 5-9
  - precedence of operations, 5-12
- EXTEND\_QUANTITY
  - SET RMS\_DEFAULT, DCL-154
- /EXTENSION
  - APPEND, DCL-10
  - COPY, DCL-26
  - INITIALIZE, DCL-64
  - SET FILE, DCL-130
  - SET VOLUME, DCL-165
- /EXTENT
  - RUN, DCL-113
  - SET WORKING\_SET, DCL-167
- /EXTRACT
  - LIBRARY, DCL-76
- EXTRACT/MAIL keypad command (MAIL), MAIL-39
- EXTRACT command (MAIL), 1-55, MAIL-13
- EXTRACT keypad command (MAIL), MAIL-39

## F

- .F (DSR command), DSR-21
- F\$CVSI lexical function, LEX-3
- F\$CVTIME lexical function, LEX-3
- F\$CVUI lexical function, LEX-4
- F\$DIRECTORY lexical function, LEX-5
- F\$EDIT lexical function, LEX-5
- F\$ELEMENT lexical function, 6-23, LEX-5
- F\$ENVIRONMENT lexical function, 1-49, 6-31, LEX-6
- F\$EXTRACT lexical function, 6-22, 6-28, LEX-8
- F\$FAO lexical function, 5-4, LEX-8
- F\$FILE\_ATTRIBUTES lexical function, LEX-11
- F\$GETDVI lexical function, LEX-13
- F\$GETJPI lexical function, 6-30, LEX-18
- F\$GETSYI lexical function, LEX-20
- F\$IDENTIFIER lexical function, LEX-21
- F\$INTEGER lexical function, LEX-22
- F\$LENGTH lexical function, LEX-22
- F\$LOCATE lexical function, LEX-23
- F\$MESSAGE lexical function, LEX-23
- F\$MODE lexical function, LEX-24
- F\$PARSE lexical function, LEX-24
- F\$PID lexical function, LEX-25
- F\$PRIVILEGE lexical function, LEX-26
- F\$PROCESS lexical function, LEX-26
- F\$SEARCH lexical function, 6-11, LEX-27
- F\$SETPRV lexical function, LEX-28
- F\$STRING lexical function, LEX-28
- F\$TIME lexical function, LEX-29
- F\$TRNLNM lexical function, LEX-29
- F\$TYPE lexical function, LEX-30
- F\$USER lexical function, LEX-31
- F\$VERIFY lexical function, LEX-31
- /FACILITY
  - SET MESSAGE, DCL-135
- /FALLBACK
  - SET PRINTER, DCL-137
  - SET TERMINAL, DCL-158
- FAO
  - parameters, LEX-11
- /FAST
  - BACKUP, DCL-17



## Index-20

- /FEED
  - PRINT, DCL-97
  - SET QUEUE/ENTRY, DCL-148
- /FF
  - SET PRINTER, DCL-137
- .FG (DSR command), DSR-21
- .FGD (DSR command), DSR-21
- FIELD command (ACL editor command), ACL-9
- .FIGURE (DSR command), DSR-21
- .FIGURE DEFERRED (DSR command), DSR-21
- File
  - carriage control, 2-13
  - characteristics, 2-12, 2-13
  - creating, 2-14
  - creating in command procedure, 6-10
  - deleting, 2-17
  - directory, 2-3
  - displaying, 2-17, 3-3
  - displaying in command procedure, 6-10
  - editing in command procedure, 6-13
  - indexed, 2-12
  - logical name, 2-20
  - merging, 2-45
  - modifying, 2-16
  - open count, 6-30
  - organization, 2-12
  - printing, 2-32
  - purging, 2-18
  - reading in command procedure, 6-12
  - record, 2-12
  - relative, 2-12
  - sequential, 2-12
  - size, 2-13
  - sorting, 2-37
  - system, 2-19
  - transfer between systems, 2-52
  - writing in command procedure, 6-10
- /FILE\_HEADER
  - DUMP, DCL-55
- /FILE\_ID
  - DIRECTORY, DCL-51
- /FILE\_LIMIT
  - RUN, DCL-113
- /FILE\_PROTECTION
  - INITIALIZE, DCL-64
  - SET VOLUME, DCL-165
- FILE command (MAIL), 1-57, MAIL-14
- FILE keypad command (MAIL), MAIL-39
- File protection, 7-11
- /FILES
  - SHOW DEVICES, DCL-169
  - SHOW MEMORY, DCL-174
  - SHOW QUEUE, DCL-177
- File specification, 2-3, 2-42
  - See also Device
  - default, 2-4, 2-5
  - foreign, 2-52
  - format, 2-4
  - in parameter list, 2-5
  - node name, 2-52
  - wildcard, 2-6
- .FILL (DSR command), DSR-21
- FILL command (EDT), 3-27, 3-28, EDT-8, EDT-30
- Filling text
  - DSR, 4-4
- FIND command (EDT), EDT-16
- FIND command (ACL editor command), ACL-9
- FIND command (EDT), 3-20, EDT-8, EDT-30
- FIRST/EDIT keypad command (MAIL), MAIL-40
- FIRST command (MAIL), MAIL-15
- FIRST keypad command (MAIL), MAIL-40
- .FIRST TITLE (DSR command), DSR-21
- .FL (DSR command), DSR-22
- .FL ACCEPT (DSR command), DSR-21
- /FLAG
  - PRINT, DCL-97
  - SET QUEUE/ENTRY, DCL-149
- Flag
  - DSR, 4-3, DSR-45
- .FLAGS ACCEPT (DSR command), DSR-21
- .FLAGS ALL (DSR command), DSR-22
- .FLAGS BOLD (DSR command), DSR-22
- .FLAGS BREAK (DSR command), DSR-22
- .FLAGS CAPITALIZE (DSR command), DSR-22



- .FLAGS COMMENT (DSR command), DSR-22
- .FLAGS CONTROL (DSR command), DSR-22
- .FLAGS HYPHENATE (DSR command), DSR-23
- .FLAGS INDEX (DSR command), DSR-23
- .FLAGS LOWERCASE (DSR command), DSR-23
- .FLAGS OVERSTRIKE (DSR command), DSR-23
- .FLAGS PERIOD (DSR command), DSR-23
- .FLAGS SPACE (DSR command), DSR-23
- .FLAGS SUBINDEX (DSR command), DSR-24
- .FLAGS SUBSTITUTE (DSR command), DSR-24
- .FLAGS UNDERLINE (DSR command), DSR-24
- .FLAGS UPPERCASE (DSR command), DSR-24
- .FL BOLD (DSR command), DSR-22
- .FL BREAK (DSR command), DSR-22
- .FL CAPITALIZE (DSR command), DSR-22
- .FL COMMENT (DSR command), DSR-22
- .FL CONTROL (DSR command), DSR-22
- .FL HYPHENATE (DSR command), DSR-23
- .FL INDEX (DSR command), DSR-23
- .FL LOWERCASE (DSR command), DSR-23
- .FL OVERSTRIKE (DSR command), DSR-23
- .FL PERIOD (DSR command), DSR-23
- .FL SPACE (DSR command), DSR-23
- .FL SUBINDEX (DSR command), DSR-24
- .FL SUBSTITUTE (DSR command), DSR-24
- .FL UNDERLINE (DSR command), DSR-24
- .FL UPPERCASE (DSR command), DSR-24
- .FN (DSR command), DSR-24
- FNDNXT command (ACL editor command), ACL-10
- FNDNXT command (EDT), 3-22, EDT-8
- Footnote
  - DSR, 4-14
- .FOOTNOTE (DSR command), DSR-24
- /FOREIGN
  - MOUNT, DCL-89
- Foreign command, 1-33
  - defining, 1-33
  - invoking, 1-34
  - precedence, 1-34
  - symbol, 5-21
- /FORM
  - PRINT, DCL-97
  - SET QUEUE/ENTRY, DCL-149
  - SET TERMINAL, DCL-158
- /FORM\_MOUNTED
  - INITIALIZE/QUEUE, DCL-70
  - SET QUEUE, DCL-145
  - START/QUEUE, DCL-195
- /FORM\_SIZE
  - RUNOFF, DSR-3
- /FORMAT
  - MERGE, DCL-80
  - SEARCH, DCL-118
  - SORT, DCL-183
- Format
  - DSR command, DSR-10
- /FORMATTED
  - DUMP, DCL-55
- Formatting
  - of DIFFERENCES output, DCL-48
- Formatting text
  - DSR, 4-3
- Form feed
  - DSR output, 4-23
- FORTRAN carriage control, 2-13
- /FORWARD
  - START/QUEUE, DCL-195
- FORWARD/EDIT keypad command (MAIL), MAIL-40
- FORWARD command (MAIL), MAIL-15
- FORWARD keypad command (MAIL), MAIL-40
- /FRAME
  - SET TERMINAL, DCL-159
- Free-page list, 1-37
- .FT (DSR command), DSR-21
- /FULL
  - ACCOUNTING, DCL-2
  - BACKUP, DCL-16, DCL-17
  - DIRECTORY, DCL-51
  - LIBRARY, DCL-76

## Index-22

### /FULL (cont'd.)

- LOGOUT, DCL-78
- SHOW ERROR, DCL-170
- SHOW LOGICAL, DCL-172
- SHOW MEMORY, DCL-174
- SHOW QUEUE, DCL-177
- SHOW QUEUE/FORM, DCL-178
- SHOW SYSTEM, DCL-180

### /FULLDUP

- SET TERMINAL, DCL-159

### Function keys

- F12, 1-24
- F14, 1-24
- VT200 terminal, 1-7

## G

### /GENERATE

- SET PASSWORD, DCL-136

### /GENERIC

- ALLOCATE, DCL-8
- INITIALIZE/QUEUE, DCL-70
- START/QUEUE, DCL-195

### Generic queue, 2-36

### /GLOBAL

- DELETE/SYMBOL, DCL-44
- INQUIRE, DCL-74
- SHOW SYMBOL, DCL-180

### /GLOBAL\_BUFFER

- SET FILE, DCL-131

### /GLOBALS

- LIBRARY, DCL-76

### GLOBALS

- LIBRARY option, DCL-74

### Global symbol, 5-16

- command procedure, 6-8

### GOLD command (ACL editor command), ACL-10

### GOLD command (EDT), 3-5, EDT-9

### GOLD keypad command (MAIL), MAIL-41

### GOSUB command, 6-23, DCL-60

### GOTO command, 6-19, DCL-60

### /GRAND\_TOTAL

- DIRECTORY, DCL-51

### Graphic character set, ESC-3

### Graphic symbol

- VT100 terminals, CHAR-5

- VT200 terminals, CHAR-5

### /GROUP

- ASSIGN, DCL-12
- DEASSIGN, DCL-32
- DEFINE, DCL-35
- INITIALIZE, DCL-64
- MOUNT, DCL-89
- SHOW LOGICAL, DCL-173

### Group

- ownership category, 7-4

### /GROUP\_SIZE

- BACKUP, DCL-17

### Group number

- in user identification code, 7-3

## H

### /HALFDUP

- SET TERMINAL, DCL-159

### /HANGUP

- LOGOUT, DCL-78
- SET TERMINAL, DCL-159

### /HARDCOPY

- SET TERMINAL, DCL-159

### .HD (DSR command), DSR-26

### .HD LOWER (DSR command), DSR-25

### .HD MIXED (DSR command), DSR-26

### /HDR3

- MOUNT, DCL-89

### .HD UPPER (DSR command), DSR-26

### /HEADER

- DUMP, DCL-55

- PRINT, DCL-98

- SET QUEUE/ENTRY, DCL-149

### .HEADER LEVEL (DSR command), DSR-25

### /HEADERS

- INITIALIZE, DCL-64

### .HEADERS [ON] (DSR command), DSR-26

### .HEADERS LOWER (DSR command), DSR-25

### .HEADERS MIXED (DSR command), DSR-26

### .HEADERS UPPER (DSR command), DSR-26

- /HEADING
  - DIRECTORY, DCL-51
  - SEARCH, DCL-118
- /HELP
  - LIBRARY, DCL-76, DCL-77
- Help
  - in interactive command, 1-23
- HELP command, 1-26, DCL-60
- HELP command (ACL editor command), ACL-11
- HELP command (EDT), 3-7, EDT-9, EDT-16, EDT-31
- HELP command (MAIL), MAIL-16
- HELP FMT command (ACL editor command), ACL-11
- HELP keypad command (MAIL), MAIL-41
- Help library, 2-46
  - creating, 2-52
  - module levels, 2-47
- /HEXADECIMAL
  - DEPOSIT, DCL-45
  - DUMP, DCL-55
  - EXAMINE, DCL-59
- /HIGHWATER
  - INITIALIZE, DCL-65
- /HIGHWATER\_MARKING
  - SET VOLUME, DCL-165
- /HISTORY
  - LIBRARY, DCL-76
- HISTORY
  - LIBRARY option, DCL-74
- .HL (DSR command), DSR-25
- HLB file type, 2-46
- HLP file type, 2-46
- /HOLD
  - PRINT, DCL-98
  - SET QUEUE/ENTRY, DCL-149
  - STOP/QUEUE, DCL-201
  - SUBMIT, DCL-205
- /HOSTSYNC
  - SET TERMINAL, DCL-159
- HYPHENATE flag, DSR-46
- I
  - .I (DSR command), DSR-27
- I/O errors
  - in command procedures, 6-16
- /IDENTIFICATION
  - ACCOUNTING, DCL-3
  - ATTACH, DCL-15
  - CANCEL, DCL-23
  - RUNOFF/CONTENTS, DSR-7
  - RUNOFF/INDEX, DSR-9
  - SET MESSAGE, DCL-135
  - SET PROCESS, DCL-139
  - SHOW PROCESS, DCL-176
  - STOP, DCL-200
- Identifier, 7-7
- IDENTIFIER access control entry, 7-8
- /IDENTIFY
  - PRINT, DCL-98
  - SUBMIT, DCL-205
- .IF (DSR command), DSR-26
- /IF\_STATE
  - DEFINE/KEY, DCL-40
- IF command, 6-19, DCL-62
- .IFNOT (DSR command), DSR-26
- /IGNORE
  - BACKUP, DCL-18
  - DIFFERENCES, DCL-47
- /IMAGE
  - ACCOUNTING, DCL-3
  - BACKUP, DCL-18
- Image, 1-37
- INCLUDE command (EDT), 3-29, EDT-31
- /INCREMENTAL
  - BACKUP, DCL-18
- /INDENT
  - RUNOFF/CONTENTS, DSR-7
- .INDENT (DSR command), DSR-27
- Indent text
  - DSR, 4-6
- /INDEX
  - INITIALIZE, DCL-65
  - READ, DCL-103
- Index
  - formatting with DSR, 4-21, DSR-47
- .INDEX (DSR command), DSR-27

## Index-24

/INDEXED  
  SET RMS\_DEFAULT, DCL-154  
/INDEXED\_SEQUENTIAL  
  MERGE, DCL-80  
  SORT, DCL-184  
Indexed sort, 2-40  
Index entry  
  DSR, 4-21  
INDEX flag, DSR-46  
Index subentry  
  DSR, 4-22  
Initialization  
  queue, 2-34  
/INITIALIZE  
  BACKUP, DCL-18  
  MOUNT, DCL-89  
INITIALIZE/QUEUE command, DCL-68  
INITIALIZE command, DCL-62  
/INPUT  
  RUN, DCL-113  
  SPAWN, DCL-191  
/INQUIRE  
  SET TERMINAL, DCL-159  
INQUIRE command, 6-6, DCL-73  
/INSERT  
  LIBRARY, DCL-76  
  SET TERMINAL, DCL-159  
INSERT command (ACL editor command),  
  ACL-11  
INSERT command (EDT), EDT-32  
INSERT HERE command (EDT), EDT-16  
/INSTRUCTIONS  
  HELP, DCL-61  
Interactive command, 1-18, 1-23  
/INTERCHANGE, DCL-18  
  BACKUP, DCL-18  
/INTERMEDIATE  
  RUNOFF, DSR-4  
/INTERVAL  
  RUN, DCL-113  
Invoking MAIL, MAIL-1  
/IO\_BUFFERED  
  RUN, DCL-113  
/IO\_DIRECT  
  RUN, DCL-114  
ITEM command (ACL editor command),  
  ACL-11

## J

.J (DSR command), DSR-27  
/JOB  
  ACCOUNTING, DCL-3  
  SHOW LOGICAL, DCL-173  
/JOB\_COUNT  
  PRINT, DCL-98  
  SET QUEUE/ENTRY, DCL-149  
/JOB\_LIMIT  
  SET QUEUE, DCL-145  
  START/QUEUE, DCL-196  
/JOB\_LIMIT/INITIALIZE/QUEUE,  
  DCL-71  
/JOB\_TABLE\_QUOTA=quota  
  RUN, DCL-114  
/JOURNAL  
  BACKUP, DCL-18  
  EDIT, EDT-2  
  EDIT/ACL, ACL-5  
  EDIT/TPU, DCL-57  
.JUSTIFY (DSR command), DSR-27  
Justify text  
  DSR, 4-4

## K

.K (DSR command), DSR-27  
/KEEP  
  PURGE, DCL-102  
  SET QUEUE/ENTRY, DCL-149  
  SUBMIT, DCL-206  
KEEP  
  LIBRARY option, DCL-74  
.KEEP (DSR command), DSR-27  
/KEY  
  MERGE, DCL-81  
  READ, DCL-103  
  SORT, DCL-184  
Key  
  sort, 2-39  
Key definition, 1-29  
  deleting, 1-31  
/KEYPAD  
  SPAWN, DCL-191

## Keypad

- ACL editor, 7-17
- Keypad commands (ACL editor), ACL-7
- Keypad commands (MAIL), 1-59, MAIL-35
- Keypad diagram (MAIL), 1-60, MAIL-35
- Keypad in EDT, 3-4, EDT-3
- KEYSIZE
  - LIBRARY option, DCL-74
- Key state, 1-30
  - changing, 1-30

## L

- /LA11
  - SET PRINTER, DCL-137
- /LA180
  - SET PRINTER, DCL-137
- /LABEL
  - BACKUP, DCL-19
  - MOUNT, DCL-89
  - SET VOLUME, DCL-165
- labels,
  - in command procedures, DCL-22
- Laser print file
  - DSR, 4-23
- LAST/EDIT keypad command (MAIL),
  - MAIL-41
- LAST command (MAIL), MAIL-17
- LAST keypad command (MAIL), MAIL-41
- .LAYOUT (DSR command), DSR-28
- .LE (DSR command), DSR-29
- .LEFT MARGIN (DSR command), DSR-28
- /LENGTH
  - DEFINE/FORM, DCL-37
- Lexical function, 5-13
  - character string manipulation, LEX-1
  - devices and files, LEX-2
  - environment, LEX-2
  - evaluating, 5-14
  - Integer/character string conversion,
    - LEX-1
  - invoking, 5-13
  - parameter list, 5-13
  - text formatting, LEX-1
  - time, LEX-2

## Lexical functions

- F\$CVSI, LEX-3
- F\$CVTIME, LEX-3
- F\$CVUI, LEX-4
- F\$DIRECTORY, LEX-5
- F\$EDIT, LEX-5
- F\$ELEMENT, 6-23, LEX-5
- F\$ENVIRONMENT, 1-49, 6-31, LEX-6
- F\$EXTRACT, 6-22, 6-28, LEX-8
- F\$FAO, 5-4, LEX-8
- F\$FILE\_ATTRIBUTES, LEX-11
- F\$GETDVI, LEX-13
- F\$GETJPI, 6-30, LEX-18
- F\$GETSYI, LEX-20
- F\$IDENTIFIER, LEX-21
- F\$INTEGER, LEX-22
- F\$LENGTH, LEX-22
- F\$LOCATE, LEX-23
- F\$MESSAGE, LEX-23
- F\$MODE, LEX-24
- F\$PARSE, LEX-24
- F\$PID, LEX-25
- F\$PRIVILEGE, LEX-26
- F\$PROCESS, LEX-26
- F\$SEARCH, 6-11, LEX-27
- F\$SETPRV, LEX-28
- F\$STRING, LEX-28
- F\$TIME, LEX-29
- F\$TRNLNM, LEX-29
- F\$TYPE, LEX-30
- F\$USER, LEX-31
- F\$VERIFY, LEX-31
- /LFFILL
  - SET TERMINAL, DCL-160
- /LIBLIST
  - HELP, DCL-61
- Librarian Utility, 2-45
- Libraries, 2-45
  - adding modules, 2-50
  - compressing, 2-51
  - creating modules, 2-47
  - deleting, 2-50
  - deleting modules, 2-51
  - displaying, 2-49
- /LIBRARY
  - HELP, DCL-61

## Index-26

### /LIBRARY (cont'd.)

INITIALIZE/QUEUE, DCL-71

START/QUEUE, DCL-196

LIBRARY command, 2-45, 2-46, DCL-74

### /LIKE

SET/ACL, ACL-17

SET DEVICE/ACL, ACL-19

SET DIRECTORY/ACL, ACL-21

SET FILE/ACL, ACL-23

### /LIMIT

SET WORKING\_SET, DCL-167

### /LINE\_EDITING

SET TERMINAL, DCL-160

LINE command (EDT), 3-12, EDT-9

Line-editing commands in EDT

See EDT commands

LINE FEED (F13), ACL-14

LINEFEED key (F13), EDT-15

### /LINES\_PER\_PAGE

RUNOFF/INDEX, DSR-9

### /LIST

BACKUP, DCL-19

LIBRARY, DCL-76

### List

DSR, 4-9

.LIST (DSR command), DSR-29

.LIST ELEMENT (DSR command), DSR-29

### Literal

passing to command procedure, 6-4

.LITERAL (DSR command), DSR-30

### Literal text

DSR, 4-8

.LM (DSR command), DSR-28

LNM\$GROUP, 2-26

LNM\$PROCESS, 2-26

LNM\$PROCESS\_DIRECTORY, 2-26

LNM\$PROCESS\_TABLE, 2-26

.LO (DSR command), DSR-28

### /LOCAL

DELETE/SYMBOL, DCL-44

INQUIRE, DCL-74

SHOW SYMBOL, DCL-180

### /LOCAL\_ECHO

SET TERMINAL, DCL-160

Local symbol, 5-16

### /LOCK\_STATE

DEFINE/KEY, DCL-40

### /LOG

ACCOUNTING, DCL-3

ALLOCATE, DCL-8

APPEND, DCL-11

ASSIGN, DCL-13

BACKUP, DCL-19

CLOSE, DCL-23

COPY, DCL-27

CREATE, DCL-28

CREATE/DIRECTORY, DCL-29

DEFINE, DCL-35

DEFINE/KEY, DCL-40

DELETE, DCL-42

DELETE/KEY, DCL-43

DELETE/SYMBOL, DCL-44

LIBRARY command, DCL-77

PURGE, DCL-102

RENAME, DCL-106

REPLY, DCL-108

RUNOFF, DSR-4

RUNOFF/CONTENTS, DSR-7

RUNOFF/INDEX, DSR-9

SEARCH, DCL-118

SET ACL, ACL-17

SET DAY, DCL-125

SET DEVICE, DCL-126

SET DEVICE/ACL, ACL-19

SET DIRECTORY, DCL-128

SET DIRECTORY/ACL, ACL-21

SET FILE, DCL-131

SET FILE/ACL, ACL-23

SET HOST, DCL-132

SET KEY, DCL-133

SET MAGTAPE, DCL-134

SET PRINTER, DCL-137

SET PROTECTION, DCL-142

SET VOLUME, DCL-165

SET WORKING\_SET, DCL-167

SHOW SYMBOL, DCL-180

SPAWN, DCL-191

UNLOCK, DCL-210

### /LOG\_FILE

SET QUEUE/ENTRY, DCL-149

SUBMIT, DCL-206



- Log file
  - batch job, 1-46
- /LOGICAL \_NAMES
  - SPAWN, DCL-191
- Logical data
  - expression, 5-11
- Logical name, 2-20
  - access mode, 2-30
  - concealed, 2-24
  - deassigning, 2-22
  - defining, 2-20, 2-21
  - device, 2-22
  - displaying, 2-22
  - duplicating, 2-22
  - equivalence name, 2-21
  - for network, 2-52
  - nontranslation, 2-24
  - precedence, 2-28
  - process logical name table directory, 2-26
  - process permanent, 2-31
  - process-private, 2-25
  - redefining, 2-31
  - scope, 2-25
  - search lists, 2-24
  - shareable, 2-25
  - system, 2-30
  - system created, 2-31
  - system directory table, 2-26
  - system permanent, 2-31
  - table creation, 2-29
  - table directories, 2-26
  - translation, 2-23
- Logical operators, 5-8
- Login
  - automatic, 1-2
  - dialin, 1-2
  - manual, 1-1
  - network, 1-5
- Login command procedure, 1-2
- /LOGOUT
  - CONNECT, DCL-24
- Logout, 1-6
  - network, 1-6
- LOGOUT command, 1-6, 1-41, DCL-78

- /LOGSOFT
  - SET MAGTAPE, DCL-134
- /LONGWORD
  - DEPOSIT, DCL-46
  - DUMP, DCL-55
  - EXAMINE, DCL-59
- Longword, 5-1
- Lost edits
  - recovery, 3-9
- /LOWERCASE
  - PRINT, DCL-98
  - SET PRINTER, DCL-137
  - SET QUEUE/ENTRY, DCL-150
  - SET TERMINAL, DCL-160
- LOWERCASE flag, DSR-46
- /LP11
  - SET PRINTER, DCL-137
- .LS (DSR command), DSR-29
- .LT (DSR command), DSR-30

## M

- /MACRO
  - LIBRARY, DCL-77
- /MAGTAPE
  - SET RMS\_DEFAULT, DCL-154
- MAIL, MAIL-1
  - creating folders, 1-57
  - creating mail files, 1-58
  - CTRL/Z, 1-50
  - defining keys, MAIL-7
  - deleting, 1-55
  - deleting folders, 1-57
  - deleting mail files, 1-58
  - displaying folders, 1-58
  - distribution list, 1-52
  - invoking, 1-49
  - keypad commands, 1-59
  - keypad diagram, MAIL-35
  - network, 1-52
  - new messages, 1-54
  - old messages, 1-54
  - organizing, 1-56
  - protection, 7-12
  - reading, 1-53

## Index-28

### MAIL (cont'd.)

- SEARCH command, 1-54
- selecting folders, 1-58
- sending, 1-49
- sending a file, 1-51
- setting default editor, 1-51
- specifying mail files, 1-58
- wastebasket folder, 1-56

### /MAILBOX

- RUN, DCL-114

- MAIL command, 1-49, 2-53, MAIL-1, MAIL-17

### MAIL commands

- ANSWER, MAIL-3
- ATTACH, MAIL-4
- BACK, MAIL-4
- COMPRESS, MAIL-5
- COPY, 1-57, MAIL-6
- CURRENT, MAIL-7
- DEFINE/KEY, MAIL-7
- DELETE, MAIL-9
- DIRECTORY, 1-58, MAIL-10
- EDIT, MAIL-11
- ERASE, MAIL-13
- EXIT, MAIL-13
- EXTRACT, 1-55, MAIL-13
- FILE, 1-57, MAIL-14
- FIRST, MAIL-15
- FORWARD, MAIL-15
- HELP, MAIL-16
- LAST, MAIL-17
- MAIL, MAIL-17
- MOVE, 1-57, MAIL-18
- NEXT, MAIL-19
- PRINT, MAIL-19
- PURGE, MAIL-20
- QUIT, MAIL-21
- READ, MAIL-21
- REPLY, MAIL-22
- SEARCH, MAIL-23
- SELECT, MAIL-24
- SEND, 1-50, MAIL-25
- SET AUTO\_PURGE, MAIL-26
- SET COPY\_SELF, MAIL-26
- SET FILE, MAIL-27
- SET FOLDER, MAIL-28

### MAIL commands (cont'd.)

- SET FORWARD, MAIL-28
- SET MAIL\_DIRECTORY, MAIL-29
- SET PERSONAL\_NAME, MAIL-30
- SET WASTEBASKET\_NAME, MAIL-30
- SHOW ALL, MAIL-30
- SHOW AUTO\_PURGE, MAIL-31
- SHOW COPY\_SELF, MAIL-31
- SHOW DELETED, MAIL-31
- SHOW FILE, MAIL-31
- SHOW FOLDER, MAIL-32
- SHOW FORWARD, MAIL-32
- SHOW KEY, MAIL-32
- SHOW MAIL\_DIRECTORY, MAIL-33
- SHOW NEW\_MAIL\_COUNT, MAIL-33
- SHOW PERSONAL\_NAME, MAIL-33
- SHOW WASTEBASKET\_NAME, MAIL-33
- SPAWN, MAIL-33

### MAIL keypad commands, MAIL-35

- BACK, MAIL-36
- BACK/EDIT, MAIL-36
- CURRENT, MAIL-36
- CURRENT/EDIT, MAIL-37
- DELETE/NOTERMINATE, MAIL-37
- diagram of keypad, MAIL-35
- DIRECTORY, MAIL-37
- DIRECTORY/FOLDER, MAIL-37
- DIRECTORY/NEW, MAIL-38
- DIRECTORY/START=99999, MAIL-38
- DIRECTORY MAIL, MAIL-37
- ENTER, MAIL-38
- ERASE/NOECHO, MAIL-39
- EXTRACT, MAIL-39
- EXTRACT/MAIL, MAIL-39
- FILE, MAIL-39
- FIRST, MAIL-40
- FIRST/EDIT, MAIL-40
- FORWARD, MAIL-40
- FORWARD/EDIT, MAIL-40
- GOLD, MAIL-41
- HELP, MAIL-41
- LAST, MAIL-41
- LAST/EDIT, MAIL-41
- NEXT, MAIL-42
- NEXT/EDIT, MAIL-42

## MAIL keypad commands (cont'd.)

PRINT, MAIL-42  
 PRINT/PRINT/NOTIFY, MAIL-42  
 READ/NEW, MAIL-43  
 REPLY, MAIL-43  
 REPLY/EDIT, MAIL-43  
 SELECT/NOTERMINATE, MAIL-44  
 SELECT MAIL, MAIL-44  
 SEND, MAIL-44  
 SEND/EDIT, MAIL-45  
 SHOW NEW\_MAIL\_COUNT, MAIL-45

## MAIL keypad diagram, 1-60

## Mail subdirectory

using, 1-59

## MAIL utility

See MAIL, MAIL commands, MAIL  
 keypad commands

## /MANUAL

SET TERMINAL, DCL-160

## /MARGIN

DEFINE/FORM, DCL-37

## Margin adjustment

DSR, 4-6

## /MATCH

DIFFERENCES, DCL-48

READ, DCL-103

SEARCH, DCL-119

## /MAXIMUM\_DIFFERENCES

DIFFERENCES, DCL-48

## /MAXIMUM\_FILES

INITIALIZE, DCL-66

## /MAXIMUM\_WORKING\_SET

RUN, DCL-114

## Member number

in user identification code, 7-3

## /MEMORY

SHOW PROCESS, DCL-176

## Memory, 1-36

balance set, 1-36, 1-38

free-page list, 1-37

mapping, 1-36

modified-page list, 1-37

page cache, 1-37

page fault, 1-37

resident system, 1-39

swapping, 1-38

## Memory (cont'd.)

virtual, 1-36

working set, 1-37

## MERGE command, 2-45, DCL-79

## /MERGED

DIFFERENCES, DCL-48

## Merging files, 2-45

collating sequence, 2-43

record format, 2-44

sequence check, 2-45

specification file, 2-42

## /MESSAGE

MOUNT, DCL-89

## /MESSAGES

RUNOFF, DSR-4

## /MODE

DIFFERENCES, DCL-48

EDIT/ACL, ACL-5

## /MODEM

SET TERMINAL, DCL-160

## Modem (or data set), 1-2

## /MODIFIED

APPEND, DCL-11

BACKUP, DCL-19

COPY, DCL-27

DELETE, DCL-42

DIRECTORY, DCL-51

PRINT, DCL-98

PURGE, DCL-102

RENAME, DCL-106

SET DIRECTORY, DCL-128

SUBMIT, DCL-206

TYPE, DCL-209

## Modified-page list, 1-37

## /MODULE

LIBRARY, DCL-77

## MODULES

LIBRARY option, DCL-74

## /MOUNT\_VERIFICATION

MOUNT, DCL-90

SET VOLUME, DCL-166

## MOUNT command, 2-21, 7-13, DCL-86

## /MOUNTED

SHOW DEVICES, DCL-169

## MOVE command (EDT), EDT-32

## MOVE command (MAIL), 1-57, MAIL-18

## Index-30

MOVE SCREEN command (ACL editor  
command), ACL-11  
multinational collating sequence, 2-40

## N

.NAJ (DSR command), DSR-30  
/NAME  
    PRINT, DCL-98  
    SET PROCESS, DCL-139  
    SET QUEUE/ENTRY, DCL-150  
    SUBMIT, DCL-206  
Name  
    logical, 2-20  
    node, 2-52  
/NAME\_ATTRIBUTES  
    ASSIGN, DCL-13  
    DEFINE, DCL-35  
/NAMES  
    LIBRARY, DCL-77  
.NAP (DSR command), DSR-30  
.NAST (DSR command), DSR-30  
.NAT (DSR command), DSR-30  
.NCC (DSR command), DSR-30  
.ND (DSR command), DSR-30  
Nested list  
    DSR, 4-11  
Net, 2-53  
Network  
    displaying, 1-4  
    link, 1-5  
    logout, 1-6  
    mail, 1-52  
    see DECnet-VAX, 2-52  
/NETWORK\_BLOCK\_COUNT  
    SET RMS\_DEFAULT, DCL-154  
Network login, 1-5  
/NEW  
    SET ACL, ACL-17  
    SET DEVICE/ACL, ACL-19  
    SET DIRECTORY/ACL, ACL-21  
    SET FILE/ACL, ACL-23  
/NEW\_FILE  
    SET ACCOUNTING, DCL-120  
    /NEW\_VERSION  
        APPEND, DCL-11  
        BACKUP, DCL-19  
        RENAME, DCL-106  
        START/QUEUE/MANAGER, DCL-199  
/NEXT  
    START/QUEUE, DCL-196  
    STOP/QUEUE, DCL-201  
NEXT/EDIT keypad command (MAIL),  
    MAIL-42  
NEXT command (EDT), EDT-50  
NEXT command (MAIL), MAIL-19  
NEXT keypad command (MAIL), MAIL-42  
NEXT SCREEN command (EDT), EDT-17  
.NF (DSR command), DSR-31  
.NFL [ALL] (DSR command), DSR-31  
.NFL ACCEPT (DSR command), DSR-31  
.NFL BOLD (DSR command), DSR-31  
.NFL BREAK (DSR command), DSR-31  
.NFL CAPITALIZE (DSR command),  
    DSR-31  
.NFL COMMENT (DSR command), DSR-31  
.NFL CONTROL (DSR command), DSR-31  
.NFL HYPHENATE (DSR command),  
    DSR-32  
.NFL INDEX (DSR command), DSR-32  
.NFL LOWERCASE (DSR command),  
    DSR-32  
.NFL OVERSTRIKE (DSR command),  
    DSR-32  
.NFL PERIOD (DSR command), DSR-32  
.NFL SPACE (DSR command), DSR-32  
.NFL SUBINDEX (DSR command), DSR-32  
.NFL SUBSTITUTE (DSR command),  
    DSR-32  
.NFL UNDERLINE (DSR command),  
    DSR-33  
.NFL UPPERCASE (DSR command),  
    DSR-33  
.NJ (DSR command), DSR-33  
.NK (DSR command), DSR-33  
.NMAX (DSR command), DSR-34  
.NMCH (DSR command), DSR-35  
.NMLS (DSR command), DSR-36  
.NMLV (DSR command), DSR-35  
.NMPG (DSR command), DSR-36

- .NMR (DSR command), DSR-36
- .NMSPG (DSR command), DSR-37
- .NNM (DSR command), DSR-33
- .NO AUTOJUSTIFY (DSR command),  
DSR-30
- .NO AUTOPARAGRAPH (DSR command),  
DSR-30
- .NO AUTOSUBTITLE, DSR-30
- .NO AUTOTABLE (DSR command), DSR-  
30
- /NOCHECKPOINT  
SET QUEUE/ENTRY, DCL-150
- .NO CONTROL CHARACTERS (DSR  
command), DSR-30
- .NO DATE (DSR command), DSR-30
- /NODE  
ACCOUNTING, DCL-3
- Node  
name, 2-52
- /NODELETE  
SET QUEUE/ENTRY, DCL-150
- /NODIRECTORY  
SET FILE, DCL-131
- /NODMA  
SET TERMINAL, DCL-158
- .NO FILL (DSR command), DSR-31
- .NO FLAGS [ALL] (DSR command),  
DSR-31
- .NO FLAGS ACCEPT (DSR command),  
DSR-31
- .NO FLAGS BOLD (DSR command),  
DSR-31
- .NO FLAGS BREAK (DSR command),  
DSR-31
- .NO FLAGS CAPITALIZE (DSR command),  
DSR-31
- .NO FLAGS COMMENT (DSR command),  
DSR-31
- .NO FLAGS CONTROL (DSR command),  
DSR-31
- .NO FLAGS HYPHENATE (DSR command),  
DSR-32
- .NO FLAGS INDEX (DSR command),  
DSR-32
- .NO FLAGS LOWERCASE (DSR command),  
DSR-32
- .NO FLAGS OVERSTRIKE (DSR command),  
DSR-32
- .NO FLAGS PERIOD (DSR command),  
DSR-32
- .NO FLAGS SPACE (DSR command),  
DSR-32
- .NO FLAGS SUBINDEX (DSR command),  
DSR-32
- .NO FLAGS SUBSTITUTE (DSR command),  
DSR-32
- .NO FLAGS UNDERLINE (DSR command),  
DSR-33
- .NO FLAGS UPPERCASE (DSR command),  
DSR-33
- .NO JUSTIFY (DSR command), DSR-33
- .NO KEEP (DSR command), DSR-33
- Nokeypad commands (EDT), EDT-53
- /NOLOCK  
READ, DCL-103
- Noncommand image, 1-33
- Nonfile device  
protection, 7-14
- .NO NUMBER (DSR command), DSR-33
- .NO PAGING (DSR command), DSR-33
- .NO PERIOD (DSR command), DSR-33
- NO SCROLL key, 1-8
- .NO SPACE (DSR command), DSR-33
- .NO SUBTITLE (DSR command), DSR-34
- /NOTE  
PRINT, DCL-98  
SET QUEUE/ENTRY, DCL-150
- Note  
DSR, 4-13
- .NOTE (DSR command), DSR-34
- /NOTIFY  
PRINT, DCL-98  
REPLY, DCL-108  
SET QUEUE/ENTRY, DCL-150  
SPAWN, DCL-192  
SUBMIT, DCL-206
- .NPA (DSR command), DSR-33
- .NPR (DSR command), DSR-33
- .NSP (DSR command), DSR-33
- .NST (DSR command), DSR-34
- .NT (DSR command), DSR-34



## Index-32

### /NUMBER

DIFFERENCES, DCL-48

DUMP, DCL-56

.NUMBER APPENDIX (DSR command),  
DSR-34

.NUMBER CHAPTER (DSR command),  
DSR-35

Numbered list  
DSR, 4-10

.NUMBER LEVEL (DSR command), DSR-35

.NUMBER LIST (DSR command), DSR-36

.NUMBER PAGE (DSR command), DSR-36

.NUMBER RUNNING (DSR command),  
DSR-36

### /NUMBERS

SEARCH, DCL-119

.NUMBER SUBPAGE (DSR command),  
DSR-37

### /NUMERIC\_\_KEYPAD

SET TERMINAL, DCL-160

Numeric data, 5-5

expression, 5-9

fraction, 5-6

negative number, 5-5

positive number, 5-5

radix, 5-5

## O

### /OBJECT

EDIT/ACL, ACL-5

LIBRARY, DCL-77

### /OBJECT\_\_TYPE

SET ACL, ACL-17

### /OCTAL

DEPOSIT, DCL-46

DUMP, DCL-56

EXAMINE, DCL-59

### /ON

INITIALIZE/QUEUE, DCL-71

START/QUEUE, DCL-196

ON command, 6-28, DCL-93

ON CONTROL\_\_Y command, 6-30

### /ONLY

LIBRARY, DCL-77

OPEN command, 2-15, 2-21, 6-10, DCL-94

OPEN LINE command (EDT), 3-12, EDT-10

### /OPERATOR, DCL-150

PRINT, DCL-98

### /OUTPUT

ACCOUNTING, DCL-3

DIFFERENCES, DCL-49

DIRECTORY, DCL-51

DUMP, DCL-56

EDIT, EDT-2

EDIT/TPU, DCL-58

HELP, DCL-61

LIBRARY, DCL-77

RUN, DCL-114

RUNOFF, DSR-4

RUNOFF/CONTENTS, DSR-8

RUNOFF/INDEX, DSR-9

SEARCH, DCL-119

SHOW ACCOUNTING, DCL-167

SHOW AUDIT, DCL-168

SHOW BROADCAST, DCL-168

SHOW DEVICES, DCL-169

SHOW ERROR, DCL-170

SHOW INTRUSION, DCL-170

SHOW LOGICAL, DCL-173

SHOW MAGTAPE, DCL-174

SHOW MEMORY, DCL-174

SHOW NETWORK, DCL-175

SHOW PRINTER, DCL-175

SHOW PROCESS, DCL-176

SHOW QUEUE, DCL-177

SHOW QUEUE/CHARACTERISTICS,  
DCL-178

SHOW QUEUE/FORM, DCL-178

SHOW RMS\_\_DEFAULT, DCL-179

SHOW SYSTEM, DCL-180

SHOW TERMINAL, DCL-181

SHOW USERS, DCL-182

SHOW WORKING\_\_SET, DCL-182

SPAWN, DCL-192

TYPE, DCL-210

OVER ACE command (ACL editor  
command), ACL-12

### /OVERLAY

BACKUP, DCL-19

COPY, DCL-27



- /OVERLAY (cont'd.)
  - MERGE, DCL-82
  - SORT, DCL-186
- /OVERSTRIKE
  - SET TERMINAL, DCL-160
- OVERSTRIKE flag, DSR-46
- /OWNER
  - ACCOUNTING, DCL-3
  - DIRECTORY, DCL-52
- Owner
  - ownership category, 7-4
- /OWNER\_UIC
  - BACKUP, DCL-20
  - CREATE, DCL-28
  - CREATE/DIRECTORY, DCL-29
  - INITIALIZE, DCL-67
  - INITIALIZE/QUEUE, DCL-71
  - MOUNT, DCL-91
  - SET DIRECTORY, DCL-128
  - SET FILE, DCL-131
  - SET PROTECTION/DEVICE, DCL-143
  - SET QUEUE, DCL-145
  - SET VOLUME, DCL-166
  - START/QUEUE, DCL-196
- Owner identifier field
  - writing characters to, DCL-65
- Ownership
  - display, 7-14
  - object, 7-4

## P

- .P (DSR command), DSR-38
- .PA (DSR command), DSR-38
- /PAGE
  - HELP, DCL-61
  - SET PRINTER, DCL-137
  - SET TERMINAL, DCL-161
  - TYPE, DCL-210
- .PAGE (DSR command), DSR-37
- /PAGE\_COUNT
  - PRINT, DCL-99
- /PAGE\_FILE
  - RUN, DCL-114
- /PAGE\_NUMBERS
  - RUNOFF/CONTENTS, DSR-8
  - RUNOFF/INDEX, DSR-9
- /PAGE\_SETUP
  - DEFINE/FORM, DCL-37
- Page cache, 1-37
- PAGE command (EDT), 3-13, EDT-10
- Page fault, 1-37
- /PAGES
  - RUNOFF, DSR-5
  - SET QUEUE/ENTRY, DCL-150
- .PAGE SIZE (DSR command), DSR-37
- Pagination
  - DSR, 4-19
- .PAGING (DSR command), DSR-38
- Paragraph
  - formatting with DSR, 4-7
- .PARAGRAPH (DSR command), DSR-38
- /PARALLEL
  - DIFFERENCES, DCL-49
- Parameter, 1-20
  - command procedure, 6-4
  - file specification, 2-5
  - format, 1-20
  - prompt, 1-22
- Parameter qualifier, 1-21
- /PARAMETERS
  - PRINT, DCL-99
  - SET QUEUE/ENTRY, DCL-151
  - SUBMIT, DCL-206
- parameters,
  - command procedure, DCL-22
- /PARENT\_TABLE
  - CREATE\_NAME\_TABLE, DCL-31
- /PARITY
  - SET TERMINAL, DCL-161
- /PASSALL
  - PRINT, DCL-99
  - SET PRINTER, DCL-138
  - SET QUEUE/ENTRY, DCL-151
- Passing data
  - to command procedure, 6-4
- Passing parameter
  - to command procedure, 6-5
- Password, 1-1
  - changing, 1-1

## Index-34

### Password (cont'd.)

- creating, 1-1
- deleting, 1-1
- in command procedure, 1-5
- in file, 1-5

### PASTE command (EDT), 3-24, EDT-11

#### /PASTHRU

- SET TERMINAL, DCL-161

#### /PAUSE

- RUNOFF, DSR-5

#### /PENDING

- REPLY, DCL-109

### Period

- DSR command format, DSR-10

### .PERIOD (DSR command), DSR-39

### PERIOD flag, DSR-46

#### /PERMANENT

- SET TERMINAL, DCL-161
- SHOW TERMINAL, DCL-181

### .PG (DSR command), DSR-37

#### /PHYSICAL

- BACKUP, DCL-20

#### /PHYSICAL\_PAGES

- SHOW MEMORY, DCL-174

#### /POOL

- SHOW MEMORY, DCL-174

### Positional qualifier, 1-21

### .PR (DSR command), DSR-39

### PREV SCREEN command (EDT), EDT-17

#### /PRIMARY

- SET DAY, DCL-125

### PRINT/PRINT/NOTIFY keypad command (MAIL), MAIL-42

#### /PRINTALL

- SET PRINTER, DCL-138

### PRINT command, 2-33, DCL-94

### PRINT command (EDT), EDT-33

### PRINT command (MAIL), MAIL-19

#### /PRINTER

- DIRECTORY, DCL-52
- DUMP, DCL-56
- SET QUEUE/ENTRY, DCL-151
- SUBMIT, DCL-206

#### /PRINTER\_PORT

- SET TERMINAL, DCL-161

### Printing

- across the network, 2-53

- at a later time, 2-33

- defaults, 2-35

- DSR output, 4-23

- files, 2-32

- multiple copies, 2-33

### Print job, 2-32

### PRINT keypad command (MAIL), MAIL-42

### Print queue, 2-32

- controlling, 2-33, 2-34

- creating, 2-34

- deleting, 2-34

- generic, 2-36

- initializing, 2-34

- multiple, 2-36

- restrictions, 2-35

- starting, 2-34

- stopping, 2-34

- terminal, 2-36

#### /PRIORITY

- ACCOUNTING, DCL-4

- PRINT, DCL-99

- RUN, DCL-114

- SET PROCESS, DCL-139

- SET QUEUE/ENTRY, DCL-151

- STOP/QUEUE, DCL-201

- SUBMIT, DCL-206

#### /PRIVILEGES

- RUN, DCL-115

- SET PROCESS, DCL-139

- SHOW PROCESS, DCL-176

#### /PROCESS

- ACCOUNTING, DCL-4

- ASSIGN, DCL-13

- DEASSIGN, DCL-33

- DEFINE, DCL-35

- SHOW LOGICAL, DCL-173

- SHOW SYSTEM, DCL-180

- SORT, DCL-186

- SPAWN, DCL-192

### Process, 1-33

- batch, 1-34

- creating, 1-34

- detached, 1-34, 7-3

- interactive, 1-34

## Process (cont'd.)

scheduling, 1-38

states, 1-38

subprocess, 1-34

## /PROCESS\_NAME

RUN, DCL-116

Process address space, 1-36

Process logical name table directory, 2-26

## /PROCESSOR

INITIALIZE/QUEUE, DCL-71

MOUNT, DCL-91

START/QUEUE, DCL-196

Process permanent logical name, 2-31

## Program

command image, 1-33

executing, 1-33

noncommand image, 1-33

system, 1-18

user, 1-18

## /PROLOG

SET RMS\_DEFAULT, DCL-154

## /PROMPT

HELP, DCL-62

READ, DCL-103

SPAWN, DCL-192

## /PROTECTION

APPEND, DCL-11

BACKUP, DCL-20

COPY, DCL-27

CREATE, DCL-28

CREATE/DIRECTORY, DCL-29

CREATE/NAME\_TABLE, DCL-31

DIRECTORY, DCL-52

INITIALIZE, DCL-67

INITIALIZE/QUEUE, DCL-71

MOUNT, DCL-92

SET FILE, DCL-131

SET PROTECTION, DCL-142

SET QUEUE, DCL-146

SET VOLUME, DCL-166

START/QUEUE, DCL-196

## Protection, 7-1

access control list based, 7-7

copied files, 7-6

default, 7-5, 7-11

device, 7-13

## Protection (cont'd.)

directory, 7-12

disk volume, 7-13

display, 7-14

file, 7-11

format for object, 7-5

mail file, 7-12

nonfile device, 7-14

sensitive files, 7-5

user data and devices, 7-5

user identification code based, 7-3

## Protection mask, 7-5

## /PROTOCOL

SET TERMINAL, DCL-161

.PS (DSR command), DSR-37

## /PUNCTUATION

INQUIRE, DCL-74

PURGE command, 2-18, DCL-100

PURGE command (MAIL), MAIL-20

Purging files, 2-18

## Q

## Qualifier, 1-21

command, 1-21

default, 1-22

format, 1-21

parameter, 1-21

positional, 1-21

## /QUEUE

ACCOUNTING, DCL-4

PRINT, DCL-99

SUBMIT, DCL-207

SYNCHRONIZE, DCL-208

## Queue

See Also Print queue

batch, 1-43

generic, 2-36

terminal, 2-36

## /QUEUE\_LIMIT

RUN, DCL-116

QUIT command (EDT), 3-3, EDT-33

QUIT command (MAIL), MAIL-21

## /QUOTA

CREATE/NAME\_TABLE, DCL-31

/QUOTA (cont'd.)  
     MOUNT, DCL-92  
     SET WORKING\_SET, DCL-167  
 /QUOTAS  
     SHOW PROCESS, DCL-176

## R

.R (DSR command), DSR-39  
 Radix  
     numeric data, 5-5  
 Range, EDT-22  
 .RE (DSR command), DSR-39  
 /READ  
     OPEN, DCL-94  
 /READ\_CHECK  
     APPEND, DCL-11  
     COPY, DCL-27  
 /READ\_ONLY  
     EDIT, EDT-2  
     EDIT/TPU, DCL-58  
 READ/NEW keypad command (MAIL),  
     MAIL-43  
 Read access category, 7-4  
 READ command, 2-15, 6-7, 6-12, DCL-102  
 READ command (MAIL), MAIL-21  
 Reading mail, 1-53  
 /READSYNC  
     SET TERMINAL, DCL-161  
 /REBUILD  
     MOUNT, DCL-92  
     SET VOLUME, DCL-166  
 RECALL command, 1-27, 1-32, DCL-104  
 /RECORD  
     BACKUP, DCL-20  
 Record  
     creating, 2-15, 2-16  
     deleting, 2-16  
     deleting in command procedure, 6-15  
     format, 2-12  
     modifying in command procedure, 6-13  
     writing in command procedure, 6-15  
 /RECORD\_BLOCKING  
     INITIALIZE/QUEUE, DCL-71,  
         DCL-146, DCL-197  
     Record format, 2-12  
     /RECORDS  
         DUMP, DCL-56  
     /RECORDSIZE  
         MOUNT, DCL-92  
     Record sort, 2-37  
     /RECOVER  
         EDIT, EDT-2  
         EDIT/ACL, ACL-5  
         EDIT/TPU, DCL-58  
     /REGIS  
         SET TERMINAL, DCL-161  
     /REJECTED  
         ACCOUNTING, DCL-4  
     /RELATIVE  
         MERGE, DCL-82  
         SET RMS\_DEFAULT, DCL-155  
         SORT, DCL-186  
     /RELEASE  
         SET QUEUE/ENTRY, DCL-151  
     /REMAINING  
         SEARCH, DCL-119  
     /REMOTE  
         ACCOUNTING, DCL-4  
         PRINT, DCL-99  
         SUBMIT, DCL-207  
     /REMOVE  
         LIBRARY, DCL-77  
         SET FILE, DCL-131  
     REMOVE command (EDT), EDT-17  
     RENAME command, 2-14, DCL-104  
     .REPEAT (DSR command), DSR-39  
     /REPLACE  
         BACKUP, DCL-20  
         COPY, DCL-27  
         LIBRARY, DCL-77  
         SET ACL, ACL-18  
         SET DEVICE/ACL, ACL-19  
         SET DIRECTORY/ACL, ACL-21  
         SET FILE/ACL, ACL-24  
     REPLACE command (EDT), EDT-11,  
         EDT-34  
     /REPLY  
         REQUEST, DCL-110  
     REPLY/EDIT keypad command (MAIL),  
         MAIL-43

REPLY command, DCL-106  
 REPLY command (MAIL), MAIL-22  
 REPLY keypad command (MAIL), MAIL-43  
 /REPORT  
     ACCOUNTING, DCL-4  
 .REQ (DSR command), DSR-39  
 REQUEST command, DCL-110  
 /QUEUE  
     SET QUEUE/ENTRY, DCL-151  
     STOP/QUEUE, DCL-201  
 /REQUIRE  
     RUNOFF/INDEX, DSR-9  
 .REQUIRE (DSR command), DSR-39  
 RESEQUENCE command (EDT), EDT-34  
 /RESERVE  
     RUNOFF/INDEX, DSR-9  
 /RESET  
     STOP/QUEUE, DCL-201  
 RESET command (EDT), 3-24, EDT-11  
 /RESOURCE\_WAIT  
     RUN, DCL-116  
     SET PROCESS, DCL-140  
 /RESTART  
     PRINT, DCL-99  
     SET QUEUE/ENTRY, DCL-151  
     START/QUEUE/MANAGER, DCL-199  
     SUBMIT, DCL-207  
 .RESTORE (DSR command), DSR-39  
 Restoring  
     lost edits, 3-9  
 /RESUME  
     SET PROCESS, DCL-141  
 /RETAIN  
     INITIALIZE/QUEUE, DCL-72  
     SET QUEUE, DCL-146  
     START/QUEUE, DCL-197  
 /RETENTION  
     SET VOLUME, DCL-166  
 RETURN command, 6-23, DCL-111  
 RETURN key, EDT-15  
 /REVERSE\_EMPHASIS  
     RUNOFF, DSR-5  
 /REWIND  
     BACKUP, DCL-21  
     SET MAGTAPE, DCL-134

/RIGHT  
     RUNOFF, DSR-6  
 .RIGHT (DSR command), DSR-39  
 .RIGHT MARGIN (DSR command),  
     DSR-40  
 .RM (DSR command), DSR-40  
 Roman numerals in lists  
     DSR, 4-12  
 Root directory, 2-20  
 Routing node, 1-3  
 .RPT (DSR command), DSR-39  
 RUN command, 1-33, 1-34, DCL-111  
 RUN detached process, 1-34  
 Running head  
     DSR, 4-18  
 Runoff, 4-1, DSR-1  
 RUNOFF/CONTENTS command, DSR-1,  
     DSR-7  
 RUNOFF/INDEX command, DSR-1  
 RUNOFF command, 4-19, DSR-1

## S

.S (DSR command), DSR-42  
 .SA (DSR command), DSR-40  
 .SAVE (DSR command), DSR-40  
 /SAVE\_SET  
     BACKUP, DCL-21  
 /SCHEDULE  
     INITIALIZE/QUEUE, DCL-72  
     RUN, DCL-116  
     START/QUEUE, DCL-197  
 /SCHEDULE=SIZE  
     SET QUEUE, DCL-146  
 /SCOPE  
     SET SYMBOL, DCL-155  
     SET TERMINAL, DCL-162  
 Screen display  
     erasing, ESC-6  
     format, 1-8  
     in EDT, 3-3  
     scrolling, 1-8  
     scrolling region, ESC-7  
     width, 1-15  
 Scrolling, 1-8

## Index-38

.SDT (DSR command), DSR-40  
/SEARCH  
  START/QUEUE, DCL-197  
SEARCH command, DCL-117  
SEARCH command (MAIL), 1-54, MAIL-23  
Search list  
  logical name, 2-24  
Search list translation, 2-24  
/SECONDARY  
  SET DAY, DCL-125  
  SET PASSWORD, DCL-136  
SECT command (EDT), 3-13, EDT-12  
/SECTION  
  EDIT/TPU, DCL-58  
Section  
  DSR, 4-16  
/SECTION\_NUMBERS  
  RUNOFF/CONTENTS, DSR-8  
/SECURE\_SERVER  
  SET TERMINAL, DCL-162  
Secure user environment, 2-32  
/SECURITY  
  DIRECTORY, DCL-52  
/SELECT  
  BACKUP, DCL-21  
  DIRECTORY, DCL-52  
SELECT/NOTERMINATE keypad  
  command (MAIL), MAIL-44  
SELECT command (EDT), 3-19, EDT-12,  
  EDT-17  
SELECT command (MAIL), 1-58, MAIL-24  
/SELECTIVE\_SEARCH  
  LIBRARY, DCL-78  
SELECT MAIL keypad command (MAIL),  
  MAIL-44  
SEND/EDIT keypad command (MAIL),  
  MAIL-45  
SEND command (MAIL), 1-50, MAIL-25  
SEND keypad command (MAIL), MAIL-44  
.SEND TOC (DSR command), DSR-40  
/SEPARATE  
  INITIALIZE/QUEUE, DCL-72  
  SET QUEUE, DCL-146  
  START/QUEUE, DCL-197  
/SEPARATE\_UNDERLINE  
  RUNOFF, DSR-6  
/SEPARATED  
  DIFFERENCES, DCL-49  
Separators  
  DSR, DSR-10  
/SEQUENCE  
  RUNOFF, DSR-6  
/SEQUENTIAL  
  MERGE, DCL-82  
  SET RMS\_DEFAULT, DCL-155  
  SORT, DCL-186  
/SERVICE\_FAILURE  
  RUN, DCL-116  
/SET\_SPEED  
  SET TERMINAL, DCL-162  
/SET\_STATE  
  DEFINE/KEY, DCL-40  
SET ACCOUNTING command, DCL-119  
SET ACL command, 7-15, 7-16, ACL-15  
SET AUDIT command, DCL-120  
SET AUTO\_PURGE command (MAIL),  
  MAIL-26  
SET AUTOREPEAT comand (EDT), EDT-34  
SET BROADCAST command, DCL-124  
SET CASE command (EDT), EDT-35  
SET COMMAND command (EDT), EDT-35  
SET CONTROL=Y command, 6-30  
SET CONTROL command, DCL-125  
SET COPY\_SELF command (MAIL),  
  MAIL-26  
SET CURSOR command (EDT), EDT-36  
.SET DATE (DSR command), DSR-40  
SET DAY command, DCL-125  
SET DEFAULT command, 2-4, DCL-126  
SET DEVICE/ACL command, 7-15, ACL-18  
SET DEVICE command, DCL-126  
SET DIRECTORY/ACL command, 7-15,  
  ACL-19  
SET DIRECTORY command, DCL-127  
SET ENTITY command (EDT), 3-34,  
  EDT-36  
SET FILE/ACL command, 7-15, ACL-22  
SET FILE command, DCL-129  
SET FILE command (MAIL), MAIL-27  
SET FNF command (EDT), EDT-37  
SET FOLDER command (MAIL), MAIL-28



- SET FORWARD command (MAIL),  
MAIL-28
- SET HELP command (EDT), EDT-37
- SET HOST command, 1-5, DCL-132
- SET KEY command, DCL-132
- SET KEYPAD command (EDT), EDT-37
- .SET LEVEL (DSR command), DSR-41
- SET LINES command (EDT), 3-33, EDT-38
- SET LOGINS/INTERACTIVE command,  
DCL-133
- SET MAGTAPE command, DCL-133
- SET MAIL\_DIRECTORY command (MAIL),  
MAIL-29
- SET MESSAGE command, 1-49, DCL-134
- SET MODE command (EDT), 3-34, EDT-38
- SET NUMBERS command (EDT), 3-19,  
EDT-38
- SET ON command, 6-29, DCL-135
- SET OUTPUT\_RATE command, DCL-135
- .SET PARAGRAPH (DSR command),  
DSR-41
- SET PASSWORD command, 1-1, DCL-136
- SET PERSONAL\_NAME command  
(MAIL), MAIL-30
- SET PRINTER command, DCL-136
- SET PROCESS command, DCL-138
- SET PROMPT command, DCL-141
- SET PROTECTION/DEFAULT command,  
DCL-142
- SET PROTECTION/DEVICE command,  
7-4, DCL-142
- SET PROTECTION command, 7-11, 7-12,  
DCL-141
- SET QUEUE/ENTRY command, DCL-147
- SET QUEUE command, DCL-143
- SET QUIET command (EDT), 3-34, EDT-38
- SET REPEAT command (EDT), EDT-39
- SET RESTART\_VALUE command,  
DCL-152
- SET RIGHTS\_LIST  
DCL commands, 7-9
- SET RIGHTS\_LIST command, DCL-153
- SET RMS\_DEFAULT command, DCL-154
- SET SCREEN command (EDT), 3-33, 3-34,  
EDT-39
- SET SEARCH command (EDT), 3-20, 3-24,  
EDT-39
- SET SUMMARY command (EDT), EDT-40
- SET SYMBOL command, 5-18, DCL-155
- SET TAB command (EDT), EDT-40
- SET TERMINAL command, 1-7, 1-8, 1-41,  
DCL-155
- SET TERMINAL command (EDT), EDT-41
- SET TEXT command (EDT), EDT-41
- .SET TIME (DSR command), DSR-42
- SET TIME command, DCL-163
- Setting keypad characteristics, ESC-8
- SET TRUNCATE command (EDT), 3-33,  
EDT-42
- SET UIC command, 7-3, DCL-164
- /SETUP  
DEFINE/FORM, DCL-37  
PRINT, DCL-99  
SET QUEUE/ENTRY, DCL-151
- Set-up  
changing terminal attributes, 1-12  
permanently changing terminal  
attributes, 1-13, 1-14  
terminal, 1-10
- SET VERIFY command, 6-26, DCL-164
- SET VERIFY command (EDT), EDT-42
- SET VOLUME command, DCL-164
- SET WASTEBASKET\_NAME command  
(MAIL), MAIL-30
- SET WORD DELIMITER command (EDT),  
EDT-42
- SET WORKING\_SET command, DCL-167
- SET WRAP command (EDT), 3-34, EDT-42
- /SEVERITY  
SET MESSAGE, DCL-135
- /SHARE  
INITIALIZE, DCL-67  
LIBRARY, DCL-77, DCL-78  
MOUNT, DCL-93  
OPEN, DCL-94
- /SHEET\_FEED  
DEFINE/FORM, DCL-38
- SHOW ACCOUNTING command,  
DCL-167
- SHOW ACL command, 7-14
- SHOW ALL command (MAIL), MAIL-30

- SHOW AUDIT command, DCL-168
- SHOW AUTO\_PURGE ocmmand (MAIL),  
MAIL-31
- SHOW AUTOREPEAT command (EDT),  
EDT-43
- SHOW BROADCAST command, DCL-168
- SHOW BUFFER command (EDT), 3-31,  
EDT-43
- SHOW CASE command (EDT), EDT-43
- SHOW COMMAND command (EDT),  
EDT-44
- SHOW COPY\_SELF command (MAIL),  
MAIL-31
- SHOW CURSOR command (EDT), EDT-44
- SHOW DEFAULT command, DCL-168
- SHOW DELETED command (MAIL),  
MAIL-31
- SHOW DEVICES command, 7-14, DCL-168
- SHOW ENTITY command (EDT), EDT-44
- SHOW ERROR command, DCL-170
- SHOW FILE command (MAIL), MAIL-31
- SHOW FILES command (EDT), EDT-44
- SHOW FNF command (EDT), EDT-45
- SHOW FOLDER command (MAIL),  
MAIL-32
- SHOW FORWARD command (MAIL),  
MAIL-32
- SHOW HELP command (EDT), EDT-45
- SHOW INTRUSION command, DCL-170
- SHOW KEY command, 1-31, DCL-171
- SHOW KEY command (EDT), EDT-45
- SHOW KEY command (MAIL), MAIL-32
- SHOW KEYPAD command (EDT), EDT-45
- SHOW LINES command (EDT), EDT-46
- SHOW LOGICAL command, 2-22,  
DCL-172
- SHOW MAGTAPE command, DCL-173
- SHOW MAIL\_DIRECTORY command  
(MAIL), MAIL-33
- SHOW MEMORY command, 1-36,  
DCL-174
- SHOW MODE command (EDT), EDT-46
- SHOW NETWORK command, 1-3,  
DCL-175
- SHOW NEW\_MAIL\_COUNT command  
(MAIL), MAIL-33
- SHOW NEW\_MAIL\_COUNT keypad  
command (MAIL), MAIL-45
- SHOW NUMBERS command (EDT),  
EDT-46
- SHOW PERSONAL\_NAME command  
(MAIL), MAIL-33
- SHOW PRINTER command, DCL-175
- SHOW PROCESS command, 1-35, 7-14,  
DCL-175
- SHOW PROTECTION command, 7-14,  
DCL-177
- SHOW QUEUE/CHARACTERISTICS  
command, DCL-178
- SHOW QUEUE/FORM command,  
DCL-178
- SHOW QUEUE command, 2-32, DCL-177
- SHOW QUIET command (EDT), EDT-46
- SHOW QUOTA command, DCL-178
- SHOW REPEAT command (EDT), EDT-47
- SHOW RMS\_DEFAULT command,  
DCL-179
- SHOW SCREEN command (EDT), EDT-47
- SHOW SEARCH command (EDT), EDT-47
- SHOW STATUS command, 1-36, DCL-179
- SHOW SUMMARY command (EDT),  
EDT-47
- SHOW SYMBOL command, 6-27, DCL-179
- SHOW SYSTEM command, 1-38, DCL-180
- SHOW TAB command (EDT), EDT-48
- SHOW TERMINAL command, 1-7,  
DCL-181
- SHOW TERMINAL command (EDT),  
EDT-48
- SHOW TEXT command (EDT), EDT-48
- SHOW TIME command, DCL-181
- SHOW TRANSLATION command,  
DCL-181
- SHOW TRUNCATE command (EDT),  
EDT-48
- SHOW USERS command, DCL-182
- SHOW VERIFY command (EDT), EDT-48
- SHOW VERSION command (EDT), EDT-49
- SHOW WASTEBASKET\_NAME command  
(MAIL), MAIL-33
- SHOW WORD command (EDT), EDT-49

SHOW WORKING\_SET command,  
     DCL-182  
 SHOW WRAP command (EDT), EDT-49  
 /SIMULATE  
     RUNOFF, DSR-6  
 /SINCE  
     ACCOUNTING, DCL-5  
     APPEND, DCL-11  
     BACKUP, DCL-21  
     COPY, DCL-27  
     DELETE, DCL-42  
     DIRECTORY, DCL-52  
     LIBRARY, DCL-78  
     PURGE, DCL-102  
     RENAME, DCL-106  
     SET ACL, ACL-18  
     SET DIRECTORY, DCL-128  
     SET DIRECTORY/ACL, ACL-21  
     SET FILE, DCL-131  
         ACL, ACL-24  
     SUBMIT, DCL-207  
     TYPE, DCL-210  
 SINCE  
     PRINT, DCL-100  
 /SIXEL\_GRAPHICS  
     SET TERMINAL, DCL-162  
 /SIZE  
     DIRECTORY, DCL-52  
 /SKIP  
     SET MAGTAPE, DCL-134  
 .SKIP (DSR command), DSR-42  
 .SL (DSR command), DSR-41  
 /SLOTS  
     SHOW MEMORY, DCL-174  
 /SOFT\_CHARACTERS  
     SET TERMINAL, DCL-162  
 /SORT  
     ACCOUNTING, DCL-5  
 Sort  
     address, 2-39  
     ascending, 2-38  
     batch job, 2-41  
     character data, 2-40  
     collating sequence, 2-40, 2-43  
     conditional keys, 2-44  
     descending, 2-38

Sort (cont'd.)  
     index, 2-40  
     indexed output file, 2-41  
     key, 2-39  
     output file, 2-41  
     record, 2-37  
     record format, 2-44  
     single key, 2-38  
     specification file, 2-42  
     tag, 2-39  
     terminal input, 2-41  
     types of, 2-37  
 Sort/Merge Utility, 2-37  
 SORT command, 2-38, DCL-182  
 .SP (DSR command), DSR-43  
 /SPACE  
     PRINT, DCL-100  
     SET QUEUE/ENTRY, DCL-152  
 Space  
     formatting with DSR, 4-13  
 SPACE flag, DSR-46  
 .SPACING (DSR command), DSR-43  
 SPAWN command, 1-39, DCL-190  
 SPAWN command (MAIL), MAIL-33  
 /SPECIFICATION  
     MERGE, DCL-82  
     SORT, DCL-186  
 Specification file, 2-42  
 SPECINS command (EDT), EDT-12  
 /SPEED  
     SET TERMINAL, DCL-162  
 .SPG (DSR command), DSR-44  
 Spooling, 2-34  
 .SPR (DSR command), DSR-41  
 /SQUEEZE  
     LIBRARY, DCL-78  
 .ST (DSR command), DSR-44  
 /STABLE  
     MERGE, DCL-86  
     SORT, DCL-190  
 /START  
     INITIALIZE/QUEUE, DCL-72  
 START/QUEUE/MANAGER command,  
     DCL-199  
 START/QUEUE command, DCL-192

## Index-42

### Startup

EDT command file, 3-32

### /STATE

DELETE

KEY, DCL-44

SET KEY, DCL-133

SHOW KEY, DCL-172

### /STATISTICS

MERGE, DCL-86

SEARCH, DCL-119

SORT, DCL-190

### /STATUS

ACCOUNTING, DCL-6

REPLY, DCL-109

.STC (DSR command), DSR-40

.STHL (DSR command), DSR-43

.STM (DSR command), DSR-42

### /STOCK

DEFINE/FORM, DCL-38

STOP/ABORT command, DCL-202

STOP/ENTRY command, DCL-202

STOP/QUEUE/MANAGER command,  
DCL-202

STOP/QUEUE command, 2-34, DCL-200

STOP/REQUEUE command, DCL-202

STOP command, 1-24, 6-3, 6-30, DCL-200

Storage units, 5-1

### /STRUCTURE

INITIALIZE, DCL-67

SHOW LOGICAL, DCL-173

.STYLE HEADERS (DSR command),  
DSR-43

Subcommand, 1-23

SUBINDEX flag, DSR-46

SUBMIT command, 1-44, DCL-203

.SUBPAGE (DSR command), DSR-44

### /SUBPROCESS

SHOW SYSTEM, DCL-180

Subprocess, 1-34

context, 1-42

creating, 1-39

deleting, 1-41

executing, 1-43

exiting, 1-41

### /SUBPROCESS\_LIMIT

RUN, DCL-116

### /SUBPROCESSES

SHOW PROCESS, DCL-176

SUBROUTINE command, 6-24

SUBS command (EDT), EDT-13

SUBSTITUTE command (EDT), 3-22,  
EDT-49

SUBSTITUTE flag, DSR-46

Substring substitution, 5-20

length, 5-20

offset, 5-20

.SUBTITLE (DSR command), DSR-44

### /SUMMARY

ACCOUNTING, DCL-6

### /SUPERVISOR\_MODE

ASSIGN, DCL-13

CREATE/NAME\_TABLE, DCL-31

DEASSIGN, DCL-33

DEFINE, DCL-35

### /SUSPEND

SET PROCESS, DCL-141

### /SWAPPING, DCL-141

RUN, DCL-117

Swapping, 1-38

### /SWITCH=DECNET

SET TERMINAL, DCL-162

### /SYMBOL

WRITE, DCL-211

Symbol, 1-27, 5-16

Access Control, 5-18

assignment, 5-17

character string, 5-21

Controlling access, DCL-155

creating, 5-17

defining in command procedure, 6-6

deleting, 5-17

displaying, 5-16

displaying in command procedure, 6-9

error condition, 1-48

evaluation, 5-17

for command, 1-27

foreign command, 1-33, 5-21

for log in, 1-5

global, 5-16, 5-18

local, 5-16, 5-18

passing to command procedure, 6-5

variable, 5-21

## /SYMBOLS

- SPAWN, DCL-192

- Symbol substitution, 5-18

- automatic, 5-18

- forced, 5-19

- order, 5-19

- SYNCHRONIZE command, DCL-208

- SYS\$OUTPUT

- redefining, 2-15

- SYSGEN parameters, 1-39

- /SYSPASSWORD

- SET TERMINAL, DCL-162

- /SYSTEM

- ASSIGN, DCL-13

- DEASSIGN, DCL-33

- DEFINE, DCL-35

- INITIALIZE, DCL-67

- MOUNT, DCL-93

- SET PASSWORD, DCL-136

- SET RMS\_DEFAULT, DCL-155

- SHOW DEVICES, DCL-169

- SHOW LOGICAL, DCL-173

- System

- ownership category, 7-4

- System address space, 1-36

- System logical name table directory, 2-26

- System parameter, 1-39

- System permanent logical name, 2-31

- SYSUAF.DAT, 7-1

## T

- .T (DSR command), DSR-45

- /TAB

- SET PRINTER, DCL-138

- SET TERMINAL, DCL-162

- Tab

- for terminal, 1-16

- TAB ADJUST comand (EDT), EDT-51

- TAB key, EDT-15

- /TABLE

- ASSIGN, DCL-13

- DEASSIGN, DCL-33

- DEFINE, DCL-35

- SHOW LOGICAL, DCL-173

- /TABLE (cont'd.)

- SHOW TRANSLATION, DCL-181

- /TABLE=command=table

- SPAWN, DCL-192

- Table of contents

- DSR, 4-20

- .TAB STOPS (DSR command), DSR-44

- Tag sort, 2-39

- Tape save set

- See Save set

- /TEMPORARY

- REPLY, DCL-109

- /TERMINAL

- ACCOUNTING, DCL-7

- INITIALIZE/QUEUE, DCL-72

- REPLY, DCL-109

- START/QUEUE, DCL-198

- Terminal

- application attribute, 1-7

- attribute, 1-17, ESC-7

- buffer, 1-8

- control characters, KEY-2

- control keys, KEY-1

- echo attribute, 1-7

- escape attribute, 1-9

- escape sequence, 1-8

- keypad, ESC-8

- numeric attribute, 1-7

- physical attribute, 1-10

- protection, 7-14

- screen display, 1-8

- scrolling, 1-8

- scrolling region, ESC-7

- set-up, 1-10

- special keys, KEY-1, KEY-2

- type-ahead, 1-8

- VT100, 1-7

- VT200, 1-7

- Terminal (VT200 Series)

- function keys, KEY-1

- Terminal I/O

- in command procedure, 6-7, 6-9

- Terminal queue, 2-36

- /TERMINATE

- DEFINE/KEY, DCL-40

- .TEST PAGE (DSR command), DSR-44



## Index-44

/TEXT  
LIBRARY, DCL-77  
SET MESSAGE, DCL-135  
Text  
entering, 1-7  
recalling, 1-32  
Text libraries, 2-46  
Time  
absolute, 5-14  
delta, 5-15  
system base, LEX-3  
/TIME\_LIMIT  
RUN, DCL-117  
/TIME\_OUT  
READ, DCL-103  
/TITLE  
ACCOUNTING, DCL-7  
.TITLE (DSR command), DSR-45  
TLB file type, 2-46  
/TO  
REPLY, DCL-109  
REQUEST, DCL-110  
TOGGLE 1/0 key, 1-17  
/TOP\_OF\_FILE  
START/QUEUE, DCL-198  
TOP command (ACL editor command),  
ACL-12  
TOP command (EDT), 3-13, EDT-13  
/TOTAL  
DIRECTORY, DCL-52  
.TP (DSR command), DSR-44  
/TRAILER  
PRINT, DCL-100  
SET QUEUE/ENTRY, DCL-152  
/TRAILING  
DIRECTORY, DCL-53  
/TRANSLATION\_ATTRIBUTES  
ASSIGN, DCL-13  
DEFINE, DCL-36  
/TRUNCATE  
BACKUP, DCL-21  
COPY, DCL-28  
DEFINE/FORM, DCL-38  
SET FILE, DCL-131  
SET PRINTER, DCL-138  
.TS (DSR command), DSR-44

/TTSYNC  
SET TERMINAL, DCL-163  
Turnkey account, 1-2  
TXT file type, 2-46  
/TYPE  
ACCOUNTING, DCL-7  
SHOW INTRUSION, DCL-170  
/TYPE\_AHEAD  
SET TERMINAL, DCL-163  
Type-ahead, 1-8  
TYPE command, 2-17, 6-10, DCL-208  
TYPE command (EDT), EDT-52

## U

UAF  
See User authorization file  
/UIC  
ACCOUNTING, DCL-7  
RUN, DCL-117  
UIC  
See User identification code  
UK character set, ESC-3  
UND ACE command (ACL editor  
command), ACL-12  
UND C command (ACL editor command),  
ACL-12  
UND C command (EDT), 3-17, EDT-13  
/UNDERLINE  
RUNOFF/CONTENTS, DSR-8  
/UNDERLINE\_CHARACTER  
RUNOFF, DSR-6  
UNDERLINE flag, DSR-46  
Underlining text  
DSR, 4-14  
UND L command (EDT), 3-18, EDT-14  
UND W command (ACL editor command),  
ACL-13  
UND W command (EDT), 3-17, EDT-14  
/UNIT  
DISMOUNT, DCL-54  
/UNIT\_RECORD  
SET RMS\_DEFAULT, DCL-155  
/UNKNOWN  
SET PRINTER, DCL-138



/UNKNOWN (cont'd.)  
     SET TERMINAL, DCL-163  
 /UNLOAD  
     DISMOUNT, DCL-54  
     MOUNT, DCL-93  
     SET MAGTAPE, DCL-134  
     SET VOLUME, DCL-166  
 /UNLOCK  
     SET FILE, DCL-131  
 UNLOCK command, DCL-210  
 /UPDATE  
     WRITE, DCL-211  
 /UPPERCASE  
     SET PRINTER, DCL-138  
     SET TERMINAL, DCL-163  
 UPPERCASE flag, DSR-46  
 /URGENT  
     REPLY, DCL-109  
 US (default) character set  
     ASCII, ESC-3  
 /USER  
     ACCOUNTING, DCL-8  
     PRINT, DCL-100  
     SHOW QUOTA, DCL-179  
     SUBMIT, DCL-207  
 /USER\_MODE  
     ASSIGN, DCL-14  
     CREATE/NAME\_TABLE, DCL-31  
     DEASSIGN, DCL-33  
     DEFINE, DCL-36  
 /USER\_NAME  
     INITIALIZE, DCL-67  
     SET VOLUME, DCL-166  
 User authorization file, 7-1  
 USERB\_ACL\_KEYPAD, ACL-6  
 User identification code  
     alphanumeric, 7-3  
     default protection, 7-11  
     member component, 7-3  
     numeric, 7-3  
     object, 7-4  
     process, 7-3  
     protection, 7-3  
 /USERLIBRARY  
     HELP, DCL-62

/USERNAME  
     REPLY, DCL-109  
 Utility, 1-19

## V

.VARIABLE (DSR command), DSR-45  
 /VARIANT  
     RUNOFF, DSR-7  
 VAXTPU editor  
     batch job, DCL-57  
     command file, DCL-57  
     create, DCL-57  
     journal file, DCL-57, DCL-58  
     recovering edits, DCL-58  
     section file, DCL-58  
     /RECOVER qualifier, DCL-58  
     start-up file, DCL-57  
     unsupported terminal, DCL-57  
 /VERIFIED  
     INITIALIZE, DCL-67  
 /VERIFY  
     BACKUP, DCL-21  
 /VERSION\_LIMIT  
     CREATE/DIRECTORY, DCL-29  
     SET DIRECTORY, DCL-128  
     SET FILE, DCL-131  
 /VERSIONS  
     DIRECTORY, DCL-53  
 Virtual memory  
     See Memory  
 /VOLUME  
     BACKUP, DCL-21  
     COPY, DCL-28  
     CREATE, DCL-29  
     CREATE/DIRECTORY, DCL-30  
 Volume  
     see Disk  
 Volume accessibility field  
     writing characters to, DCL-65  
 VR (DSR command), DSR-45  
 VT100 Terminal  
     See Also Terminal  
     baud rate, 1-17

**VT200 Terminal**

- See Also Terminal
- baud rate, 1-14
- Field cursor in Set-Up, 1-10
- Set-Up directory, 1-10, 1-11
- Set-Up screens, 1-10
- supplemental editing keypad, 3-5

**W**

**/WAIT**

- REPLY, DCL-109
- SPAWN, DCL-192

**WAIT command, DCL-210**

**/WIDTH**

- DEFINE/FORM, DCL-38
- DIFFERENCES, DCL-49
- DIRECTORY, DCL-53
- LIBRARY, DCL-78
- SET PRINTER, DCL-138
- SET TERMINAL, DCL-163

**Wildcard characters, 2-6**

- asterisk, 2-7
- ellipsis, 2-7
- hyphen, 2-8
- percent sign, 2-7

**/WINDOW**

- DIFFERENCES, DCL-49
- SEARCH, DCL-119

**/WINDOWS**

- INITIALIZE, DCL-67
- MOUNT, DCL-93
- SET VOLUME, DCL-166
- SHOW DEVICES, DCL-170

**/WORD**

- DEPOSIT, DCL-46
- DUMP, DCL-56
- EXAMINE, DCL-59

**Word, 5-1**

**WORD command (ACL editor command),  
ACL-13**

**WORD command (EDT), 3-11, EDT-14**

**/WORK\_FILES**

- SORT, DCL-190

**/WORKING\_SET**

- RUN, DCL-117

**Working set, 1-37**

**World**

- ownership category, 7-4

**/WRAP**

- DEFINE/FORM, DCL-38
- SET PRINTER, DCL-138
- SET TERMINAL, DCL-163

**WRAP command (EDT), 3-27**

**/WRITE**

- MOUNT, DCL-93
- OPEN, DCL-94

**/WRITE\_CHECK**

- APPEND, DCL-11
- COPY, DCL-28

**Write access category, 7-4**

**WRITE command, 2-15, 6-9, 6-11, DCL-211**

**WRITE command (EDT), 3-29, EDT-52**

**/WSDEFAULT**

- INITIALIZE/QUEUE, DCL-72
- SET QUEUE, DCL-147
- SET QUEUE/ENTRY, DCL-152
- START/QUEUE, DCL-198
- SUBMIT, DCL-207

**/WSEXTEND**

- INITIALIZE/QUEUE, DCL-73
- SET QUEUE, DCL-147
- SET QUEUE/ENTRY, DCL-152

**/WSEXTENT**

- START/QUEUE, DCL-198
- SUBMIT, DCL-207

**/WSQUOTA**

- INITIALIZE/QUEUE, DCL-73
- SET QUEUE, DCL-147
- SET QUEUE/ENTRY, DCL-152
- START/QUEUE, DCL-198
- SUBMIT, DCL-207

**X**

**.X (DSR command), DSR-27**

**.XL (DSR command), DSR-45**

**.XLOWER (DSR command), DSR-45**

**.XU (DSR command), DSR-45**

**.XUPPER (DSR command), DSR-45**

**Y**

**.Y (DSR command), DSR-20**

## 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 user/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

SSG PUBLICATIONS ZK1-3/J35  
DIGITAL EQUIPMENT CORPORATION  
110 SPIT BROOK ROAD  
NASHUA, NEW HAMPSHIRE 03062-2698



Do Not Tear - Fold Here

Cut Along Dotted Line

## 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 user/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

SSG PUBLICATIONS ZK1-3/J35  
DIGITAL EQUIPMENT CORPORATION  
110 SPIT BROOK ROAD  
NASHUA, NEW HAMPSHIRE 03062-2698



Do Not Tear - Fold Here

Cut Along Dotted Line