

# **Guide to VAX DEC/Test Manager**

Order Number: AA-Z330E-TE

**December 1989**

This manual describes the concepts, commands, features, and DECwindows interface of VAX DEC/Test Manager.

**Revision/Update Information:** This revised manual supersedes *Guide to VAX DEC/Test Manager* (Order Number AA-Z330D-TE).

**Operating System and Version:** VMS Version 5.3 or higher

**Software Version:** VAX DEC/Test Manager Version 3.1

**digital equipment corporation  
maynard, massachusetts**

---

**First Printing, November 1984**  
**Revised, December 1985**  
**Revised, October 1986**  
**Revised, May 1989**  
**Revised, December 1989**

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.


Restricted Rights: Use, duplication, or disclosure by the U.S. Government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013.

© Digital Equipment Corporation 1984, 1985, 1986, 1989.

All Rights Reserved.  
Printed in U.S.A.

The postpaid Reader's Comments forms at the end of this document request your critical evaluation to assist in preparing future documentation.

The following are trademarks of Digital Equipment Corporation:

CDD/Plus	VAX Document	VAXstation
DATATRIEVE	VAX MACRO	VMS
DECforms	VAX Notes	VT
DECwindows	VAX SCAN	
VAX	VAXcluster	
VAX DIBOL	VAXset	

ZK5331

# Contents

---

<b>Preface</b> .....	xv
----------------------	----

---

## **Chapter 1 Introduction to DEC/Test Manager**

<b>1.1 Overview</b> .....	1-1
<b>1.2 Entering Commands</b> .....	1-4
1.2.1 Getting Help .....	1-5
1.2.2 Canceling Commands .....	1-5
<b>1.3 Getting Started</b> .....	1-6

---

## **Chapter 2 Using DEC/Test Manager in DECwindows**

<b>2.1 Overview</b> .....	2-1
2.1.1 Getting Help .....	2-2
2.1.2 Displaying DEC/Test Manager Information in DECwindows .....	2-2
2.1.3 DEC/Test Manager Command Correlation .....	2-4
<b>2.2 Sample DECwindows Session</b> .....	2-4
2.2.1 Creating a Library .....	2-4
2.2.2 Creating a Test .....	2-5
2.2.3 Recording a Test .....	2-6
2.2.4 Creating a Collection .....	2-10
2.2.5 Executing a Collection .....	2-11
2.2.6 Displaying a Test Result .....	2-12
2.2.7 Updating a Benchmark File .....	2-14
2.2.8 Creating a Benchmark Mask .....	2-15

---

**Chapter 3     Creating Tests**

<b>3.1</b>	<b>DEC/Test Manager Libraries</b> .....	<b>3-1</b>
3.1.1	Creating a DEC/Test Manager Library .....	3-1
3.1.2	Setting a Library .....	3-1
3.1.3	Displaying DEC/Test Manager Library Information .....	3-1
3.1.4	DEC/Test Manager History .....	3-1
	3.1.4.1 Adding a Remark to the History .....	3-1
	3.1.4.2 Deleting History Information .....	3-1
<b>3.2</b>	<b>DEC/Test Manager Tests</b> .....	<b>3-7</b>
<b>3.3</b>	<b>Test Descriptions</b> .....	<b>3-8</b>
3.3.1	Creating Test Descriptions .....	3-8
3.3.2	Displaying Test Descriptions .....	3-10
3.3.3	Copying Test Descriptions .....	3-10
3.3.4	Modifying Test Descriptions .....	3-12
3.3.5	Deleting Test Descriptions .....	3-13
<b>3.4</b>	<b>Creating Noninteractive Tests</b> .....	<b>3-14</b>
3.4.1	Writing a Noninteractive Test .....	3-14
3.4.2	Writing a Template File for a Noninteractive Test .....	3-14
3.4.3	Creating a Noninteractive Test Description .....	3-15
<b>3.5</b>	<b>Creating Interactive Tests</b> .....	<b>3-16</b>
3.5.1	Recording Tests .....	3-17
	3.5.1.1 Recording Key Sequences .....	3-17
	3.5.1.2 Exiting from a Recording Session .....	3-19
	3.5.1.3 Redefining the Termination Character .....	3-19
	3.5.1.4 Redefining the Command Keysym Key .....	3-20
3.5.2	Interactive Terminal Recording .....	3-20
3.5.3	Interactive DECwindows Recording .....	3-21
<b>3.6</b>	<b>Creating an Input File from a Session File</b> .....	<b>3-22</b>
<b>3.7</b>	<b>Playing Back an Interactive Test</b> .....	<b>3-22</b>
3.7.1	Playing an Interactive Terminal Session .....	3-23
3.7.2	Playing an Interactive DECwindows Session .....	3-23
<b>3.8</b>	<b>Processing Considerations for Interactive Terminal Tests</b> .....	<b>3-23</b>
3.8.1	Time-Dependent Applications .....	3-24
3.8.2	CTRL/C or CTRL/Y .....	3-24
3.8.3	Type-Ahead .....	3-24
3.8.4	Applications That Accept Unsolicited Input .....	3-25
3.8.5	Device Type and Terminal Characteristics .....	3-25



<b>3.9</b>	<b>Processing Considerations for DECwindows Tests</b> . . . . .	<b>3-25</b>
3.9.1	Playing DECwindows Tests . . . . .	3-25
3.9.2	Storing DECwindows Benchmark and Result Files . . . . .	3-27
3.9.3	Environment Initialization . . . . .	3-27

---

## **Chapter 4 Organizing and Executing Test Collections**

<b>4.1</b>	<b>Creating Collections</b> . . . . .	<b>4-1</b>
<b>4.2</b>	<b>Executing Collections</b> . . . . .	<b>4-3</b>
4.2.1	Executing Collections in Batch . . . . .	4-5
4.2.2	Executing Collections Interactively . . . . .	4-5
4.2.3	Stopping the Execution of Collections . . . . .	4-6
<b>4.3</b>	<b>Displaying a Collection Summary</b> . . . . .	<b>4-7</b>
<b>4.4</b>	<b>Deleting Collections</b> . . . . .	<b>4-7</b>
<b>4.5</b>	<b>Re-creating Collections</b> . . . . .	<b>4-8</b>
<b>4.6</b>	<b>Comparing Test Results</b> . . . . .	<b>4-9</b>
<b>4.7</b>	<b>Recomparing Partially Compared Collections</b> . . . . .	<b>4-10</b>

---

## **Chapter 5 Reviewing Test Results**

<b>5.1</b>	<b>Review Concepts</b> . . . . .	<b>5-1</b>
5.1.1	Using Result Descriptions . . . . .	5-2
5.1.1.1	Output Files . . . . .	5-3
5.1.1.2	Comparison Status . . . . .	5-3
5.1.2	Specifying Result Descriptions . . . . .	5-4
<b>5.2</b>	<b>Examining Test Results</b> . . . . .	<b>5-6</b>
5.2.1	Using the Review Subsystem . . . . .	5-6
5.2.1.1	Review Subsystem Overview . . . . .	5-6
5.2.1.2	Primary and Read-only Reviewers . . . . .	5-7
5.2.1.3	Canceling Review Subsystem Commands . . . . .	5-8
5.2.1.4	Locating Test Results in the Review Subsystem . . . . .	5-8
5.2.1.5	Using the Review Subsystem Keypads . . . . .	5-10
5.2.2	Displaying Test Results . . . . .	5-16
5.2.3	Printing Test Results . . . . .	5-20

<b>5.3</b>	<b>Working with Test Results</b> .....	5-21
5.3.1	Updating an Existing Benchmark File .....	5-21
5.3.2	Creating a Benchmark File for a New Test .....	5-22
5.3.3	Reviewing Partially Run Collections .....	5-25

---

## **Chapter 6 Tailoring Your Test System**

<b>6.1</b>	<b>Using Prologue and Epilogue Files</b> .....	6-1
6.1.1	Test Prologue and Epilogue Files .....	6-2
6.1.2	Collection Prologues and Epilogues .....	6-4
<b>6.2</b>	<b>Grouping Tests</b> .....	6-6
6.2.1	Organizing Tests into Groups .....	6-7
6.2.2	Displaying a Group Structure .....	6-9
6.2.3	Removing Tests and Subgroups from Groups .....	6-10
6.2.4	Deleting Groups .....	6-10
<b>6.3</b>	<b>Using Variables</b> .....	6-11
6.3.1	Modifying and Deleting Variables .....	6-13
6.3.2	Overriding Variable Default Values .....	6-13
6.3.3	Using Variables Defined by DEC/Test Manager .....	6-15
6.3.3.1	DTM\$COLLECTION_NAME Global Symbol .....	6-15
6.3.3.2	DTM\$TEST_NAME Local Symbol .....	6-15
6.3.3.3	DTM\$RESULT Logical Name .....	6-16
6.3.3.4	DTM\$DECW\$DISPLAY Logical Name .....	6-17
6.3.3.5	DTM\$DELAY_TIMEOUT Logical Name .....	6-17
6.3.3.6	DTM\$OMIT_PRINTABLE_SCREEN Logical Name .....	6-18
<b>6.4</b>	<b>Using Filters</b> .....	6-18
6.4.1	Associating and Disabling Test Filters .....	6-19
6.4.2	Applying File Filters .....	6-20
<b>6.5</b>	<b>Defining Keypad Keys</b> .....	6-20
<b>6.6</b>	<b>Using Command Files</b> .....	6-22
6.6.1	Creating and Invoking a Command File .....	6-22
6.6.2	Creating a DEC/Test Manager Initialization Command File ...	6-23
<b>6.7</b>	<b>Spawning or Attaching to Another Process</b> .....	6-24

---

<b>Chapter 7</b>	<b>Maintaining a DEC/Test Manager Library</b>	
7.1	Correcting an Invalid DEC/Test Manager Library . . . . .	7-1
7.2	Storing Files Outside a DEC/Test Manager Library . . . . .	7-2
7.2.1	Setting Benchmark and Template Directories . . . . .	7-2
7.2.2	Storing Files in CMS Libraries . . . . .	7-3
7.3	Security Features . . . . .	7-5
7.3.1	Assigning UIC Protection . . . . .	7-5
7.3.2	Assigning ACL Protection . . . . .	7-7
7.3.2.1	Using ACLs on Library Directories . . . . .	7-7
7.3.2.2	Using ACLs on Library Files . . . . .	7-8

---

<b>Chapter 8</b>	<b>Working with Terminal Session Files</b>	
8.1	Terminal Session Files . . . . .	8-2
8.1.1	Sample Session File . . . . .	8-3
8.1.2	Terminal Session File Structure . . . . .	8-5
8.1.2.1	Record Structure of Session Files . . . . .	8-6
8.1.2.2	Modifying Session Files Directly . . . . .	8-9
8.2	Input Files . . . . .	8-11
8.2.1	Sample Input File . . . . .	8-11
8.2.2	Special Strings . . . . .	8-12
8.2.2.1	Types of Special Strings Recognized by DEC/Test Manager . . . . .	8-12
8.2.2.2	Using Special Strings in Input Files . . . . .	8-15
8.3	Creating Input Files . . . . .	8-15
8.3.1	Extracting an Input File from a Session File . . . . .	8-16
8.3.2	Creating an Input File with a Text Editor . . . . .	8-17
8.4	Recording a Session File from an Input File . . . . .	8-17
8.4.1	Using the /INPUT Qualifier . . . . .	8-18
8.4.2	Using the INSERT Recording Function . . . . .	8-19
8.4.3	Terminal Characteristics . . . . .	8-19
8.4.4	Type-Ahead . . . . .	8-21
8.5	Translation Tables . . . . .	8-21

---

**Chapter 9 Working with DECwindows Session Files**

<b>9.1</b>	<b>Creating a DECwindows Input File</b> .....	<b>9-1</b>
<b>9.2</b>	<b>Creating a DECwindows Session File from a DECwindows Input File</b> .....	<b>9-1</b>
<b>9.3</b>	<b>Editing a DECwindows Input File</b> .....	<b>9-1</b>
9.3.1	Commenting Input Files .....	9-1
9.3.2	Synchronizing Play Back .....	9-1
9.3.3	Repeating Tasks in an Input File .....	9-7
9.3.4	Creating Informational Messages .....	9-8
9.3.5	Changing the Times of Input Events .....	9-11

---

**Chapter 10 DEC/Test Manager Callable Interface**

<b>10.1</b>	<b>Calling Sequence for DTM\$DTM</b> .....	<b>10-1</b>
10.1.1	Command Line (command_line) .....	10-2
10.1.2	Message Routine (msg_routine) .....	10-2
10.1.3	Prompt Routine (prompt_routine) .....	10-2
10.1.4	Confirmation Routine (confirm_routine) .....	10-3
10.1.5	Output Routine (output_routine) .....	10-5
10.1.6	Output Width (width) .....	10-5
10.1.7	Initialization Flag (init_flag) .....	10-6
<b>10.2</b>	<b>Rules for Writing DEC/Test Manager Callback Routines</b> .....	<b>10-6</b>
<b>10.3</b>	<b>Handling Error Conditions</b> .....	<b>10-7</b>
<b>10.4</b>	<b>Writing an Error Message Handler</b> .....	<b>10-7</b>
<b>10.5</b>	<b>Linking with the DEC/Test Manager Image</b> .....	<b>10-8</b>

---

**Command Dictionary**

<b>1</b>	<b>Command Format</b> .....	<b>CD-3</b>
1.1	Command Parameters .....	CD-3
1.2	Qualifiers .....	CD-4
1.2.1	Command Qualifiers .....	CD-5
1.2.2	Parameter Qualifiers .....	CD-5
1.3	Remark .....	CD-5

<b>2</b>	<b>File Specification Format</b> .....	CD-6
<b>3</b>	<b>Command Descriptions</b> .....	CD-6
	@file-specification .....	CD-7
	ATTACH .....	CD-9
	COMPARE .....	CD-11
	CONVERT LIBRARY .....	CD-17
	COPY TEST_DESCRIPTION .....	CD-19
	CREATE COLLECTION .....	CD-24
	CREATE GROUP .....	CD-32
	CREATE LIBRARY .....	CD-34
	CREATE TEST_DESCRIPTION .....	CD-36
	CREATE VARIABLE .....	CD-43
	DEFINE/KEY .....	CD-47
	DELETE COLLECTION .....	CD-51
	DELETE GROUP .....	CD-53
	DELETE HISTORY .....	CD-56
	DELETE TEST_DESCRIPTION .....	CD-59
	DELETE VARIABLE .....	CD-62
	DISPLAY .....	CD-64
	DTM .....	CD-66
	EXIT .....	CD-68
	EXTRACT .....	CD-69
	FILTER .....	CD-72
	HELP .....	CD-75
	INSERT GROUP .....	CD-77
	INSERT TEST_DESCRIPTION .....	CD-80
	MODIFY GROUP .....	CD-83
	MODIFY TEST_DESCRIPTION .....	CD-85
	MODIFY VARIABLE .....	CD-93
	PLAY .....	CD-96
	RECORD .....	CD-99
	RECREATE .....	CD-104
	REMARK .....	CD-107
	REMOVE GROUP .....	CD-109
	REMOVE TEST_DESCRIPTION .....	CD-111
	RESTORE .....	CD-114
	REVIEW .....	CD-116
	RUN .....	CD-119
	SET BENCHMARK_DIRECTORY .....	CD-123
	SET EPILOGUE .....	CD-125
	SET LIBRARY .....	CD-127
	SET PROLOGUE .....	CD-129
	SET TEMPLATE_DIRECTORY .....	CD-131

SHOW ALL	CD-133
SHOW BENCHMARK_DIRECTORY	CD-135
SHOW COLLECTION	CD-136
SHOW EPILOGUE	CD-140
SHOW GROUP	CD-141
SHOW HISTORY	CD-144
SHOW LIBRARY	CD-148
SHOW PROLOGUE	CD-149
SHOW TEMPLATE_DIRECTORY	CD-150
SHOW TEST_DESCRIPTION	CD-151
SHOW VARIABLE	CD-156
SHOW VERSION	CD-159
SPAWN	CD-160
STOP	CD-164
SUBMIT	CD-166
VERIFY	CD-169
<b>4 Review Subsystem Command Descriptions</b>	<b>CD-171</b>
DTM_REVIEW> @file-specification	CD-172
DTM_REVIEW> ATTACH	CD-174
DTM_REVIEW> BACK	CD-176
DTM_REVIEW> DEFINE/KEY	CD-178
DTM_REVIEW> EXIT	CD-182
DTM_REVIEW> FIRST	CD-184
DTM_REVIEW> HELP	CD-185
DTM_REVIEW> INSERT	CD-187
DTM_REVIEW> LAST	CD-190
DTM_REVIEW> NEXT	CD-191
DTM_REVIEW> PCA	CD-194
DTM_REVIEW> PRINT	CD-195
DTM_REVIEW> SELECT	CD-199
DTM_REVIEW> SHOW	CD-200
DTM_REVIEW> SPAWN	CD-206
DTM_REVIEW> UPDATE	CD-210

---

## Appendix A DEC/Test Manager Messages

<b>A.1 Message Display</b>	<b>A-1</b>
A.1.1 Message Format	A-1
A.1.2 Severity Codes	A-2
<b>A.2 DEC/Test Manager Messages</b>	<b>A-2</b>

---

## Glossary

---

## Index

---

### Examples

1-1	Sample Interactive Terminal Session . . . . .	1-7
3-1	Copying Test Descriptions . . . . .	3-12
3-2	Sample Noninteractive Test File . . . . .	3-14
3-3	Sample Noninteractive Test Template File . . . . .	3-15
5-1	Updating a Benchmark File . . . . .	5-22
5-2	Creating a Benchmark File . . . . .	5-23
6-1	Sample Test Prologue File . . . . .	6-3
6-2	Sample Test Epilogue File . . . . .	6-4
6-3	Sample Collection Prologue File . . . . .	6-5
6-4	Sample Collection Epilogue File . . . . .	6-6
6-5	Creating Groups . . . . .	6-8
6-6	Overriding Variables . . . . .	6-14
6-7	Using the DTM\$TEST_NAME Local Symbol . . . . .	6-16
6-8	Using the DTM\$RESULT Logical Name . . . . .	6-16
6-9	Sample Initialization Command File . . . . .	6-23
8-1	Sample Session File . . . . .	8-4
8-2	Inserting an Input File into a Recording Session . . . . .	8-20
9-1	DECwindows Input File . . . . .	9-2
9-2	Commented Input File . . . . .	9-6
9-3	Adding Synchronization Points . . . . .	9-7
9-4	Adding Loops to an Input File . . . . .	9-8
9-5	Creating Informational Messages . . . . .	9-9

---

### Figures

1-1	Regression Testing with DEC/Test Manager . . . . .	1-2
2-1	DEC/Test Manager DECwindows Title Bar and Main Menus . . . . .	2-2
2-2	Expanded Collection View . . . . .	2-3
2-3	Creating a DEC/Test Manager Library . . . . .	2-5
2-4	Creating a DECwindows Test . . . . .	2-6
2-5	Recording a DECwindows Test . . . . .	2-7

2-6	Ready to Record Dialog Box . . . . .	2-8
2-7	Sample DECwindows Recording Session . . . . .	2-9
2-8	Creating a Collection . . . . .	2-10
2-9	Executing a Collection . . . . .	2-11
2-10	Viewing Differences . . . . .	2-13
2-11	Updating a Benchmark Image . . . . .	2-14
2-12	Applying Masks to a Benchmark Image . . . . .	2-16
3-1	Overview of a Custom DEC/Test Manager Library . . . . .	3-2
5-1	Review Subsystem Default Keypad . . . . .	5-11
5-2	Review Subsystem SHOW/RESULT, SHOW/BENCHMARK, and DISPLAY/BENCHMARK Keypad . . . . .	5-13
5-3	Review Subsystem SHOW/DIFFERENCES Default Keypad . . . . .	5-15
5-4	DEC/Test Manager Benchmark File Screen 0 . . . . .	5-18
5-5	DEC/Test Manager Difference File Screen 0 . . . . .	5-19
6-1	Sample Group Hierarchy . . . . .	6-9
8-1	Format of the Terminal Characteristics Block . . . . .	8-7

---

## Tables

1-1	Supported DEC/Test Manager Environments . . . . .	1-3
1-2	DEC/Test Manager Terms . . . . .	1-3
3-1	DEC/Test Manager Test Types . . . . .	3-7
3-2	Test Description Fields . . . . .	3-8
3-3	Recording Key Sequences . . . . .	3-18
4-1	Comparison Status Values . . . . .	4-9
5-1	DEC/Test Manager Output Files . . . . .	5-3
5-2	Result Descriptions and Comparison Status Qualifier Variations for the SHOW Command . . . . .	5-5
5-3	Locating Test Results . . . . .	5-8
5-4	Key Definitions for the Review Subsystem Keypad . . . . .	5-11
5-5	Key Definitions for the SHOW/RESULT, SHOW/BENCHMARK, and DISPLAY/BENCHMARK Keypad . . . . .	5-14
5-6	Key Definitions for the SHOW/DIFFERENCES Keypad . . . . .	5-15
6-1	Prologue and Epilogue Files . . . . .	6-2
6-2	Group Commands . . . . .	6-7
7-1	DEC/Test Manager Action in CMS Libraries . . . . .	7-3
7-2	Privileges Required for DEC/Test Manager Library Users . . . . .	7-6
7-3	Privileges Required for Individual DEC/Test Manager Commands . . . . .	7-8
8-1	Session File Record Types . . . . .	8-8



8-2	Translation of Nonprinting Characters and Control Codes When Extracting an Input File from a Session File . . . . .	8-21
8-3	Translation of Special Strings Representing Control and Nonprinting Characters When Recording a Session File from an Input File . . . . .	8-27
8-4	Translation of Special Strings Representing 8-Bit Control Characters When Recording a Session File from an Input File . . . . .	8-29
8-5	Translation of Special Strings Representing the Function Key Codes When Recording a Session File from an Input File . . . . .	8-30
8-6	Translation of Special Strings Representing the Editing Key Codes When Recording a Session File from an Input File . . . . .	8-31
8-7	Translation of Special Strings Representing the Keypad Key Codes When Recording a Session File from an Input File . . . . .	8-31
8-8	Translation of Special Strings Representing the Arrow Key Codes When Recording a Session File from an Input File . . . . .	8-33
8-9	Translation of Special Strings Representing the Recording Functions When Recording a Session File from an Input File . . . . .	8-34
9-1	Input File Editing Operations . . . . .	9-5
10-1	Confirm_Routine Response String . . . . .	10-4
10-2	Confirm Routine Return Status . . . . .	10-4
CD-1	DEC/Test Manager Command Line Elements . . . . .	CD-3
CD-2	Types of Collection Display Information . . . . .	CD-137
A-1	DEC/Test Manager Message Fields . . . . .	A-1
A-2	DEC/Test Manager Message Severity Codes . . . . .	A-2



# Preface

---

This manual explains how to use DEC/Test Manager as an automated regression test system. It describes how to use DEC/Test Manager during the development and maintenance phase of a software development project.

DEC/Test Manager allows you flexibility in organizing tests, in selecting tests for execution, and in reviewing and verifying test results. This manual provides examples that show basic and advanced techniques for using DEC/Test Manager.

---

## Intended Audience

This manual is intended primarily for programmers, software engineers, and project managers who are responsible for producing fully tested code. Users should be familiar with the VMS operating system, Digital Command Language (DCL), VMS program development facilities, and VMS utilities.

---

## Document Structure

The *Guide to VAX DEC/Test Manager* contains 10 chapters, a command dictionary, an appendix, and a glossary.

Chapter 1 explains regression testing, gives an overview of how DEC/Test Manager automates the regression testing process, and briefly explains the DEC/Test Manager concepts. It includes a sample interactive terminal session.

Chapter 2 describes the DEC/Test Manager DECwindows interface. It gives an overview and a sample DECwindows session.

Chapter 3 describes the concepts of DEC/Test Manager libraries and tests. It explains how to create noninteractive tests and how to record interactive terminal tests.

Chapter 4 explains how to organize, execute, display, delete, re-create, compare, and stop test collections.

Chapter 5 explains how to examine test results. Topics include evaluating the results of a test run, examining and updating test results, and displaying and printing reports of the test results.

Chapter 6 explains how to add features to your test system to create a custom testing environment. Topics include prologue and epilogue files, groups, variables, filters, defining keys for the DEC/Test Manager keypads, initialization and command files, and spawning subprocesses.

Chapter 7 explains how to store files outside the DEC/Test Manager library, verify data in the DEC/Test Manager library, access the DEC/Test Manager library from a remote node, and how to set user identification code (UIC) and access control list (ACL) file protections.

Chapter 8 describes the format of terminal session files and input files and explains how to edit terminal session files.

Chapter 9 describes the format of DECwindows session files and input files and explains how to edit DECwindows session files.

Chapter 10 explains how to call DEC/Test Manager from other programs.

The Command Dictionary describes the command syntax and the file specification format; it also contains detailed descriptions of all terminal-based DEC/Test Manager and Review subsystem commands, listed alphabetically by command name.

Appendix A contains the informational, warning, and error messages that DEC/Test Manager issues; it also supplies explanations and user actions, where applicable.

The Glossary defines DEC/Test Manager terms.

---

## Associated Documents

The following list describes additional documentation related to DEC/Test Manager:

- The *VAX DEC/Test Manager Installation Guide* supplies instructions for installing DEC/Test Manager on a VMS operating system.

- *Using VAXset* describes how to use VAX Software Engineering Tools (VAXset) with other VMS facilities to create an effective software development environment.
- The *Guide to VAX DEC/Code Management System* provides information about the VAX DEC/Code Management System (CMS).
- The *Guide to VAX Performance and Coverage Analyzer* provides reference information about the VAX Performance and Coverage Analyzer (PCA).

---

## Conventions

The following conventions are used in this guide:

Conventions	Description
<b>DELETE</b>	A key name is shown enclosed to indicate that you press a key on the keyboard.
<b>CTRL/x</b>	A sequence such as CTRL/x indicates that you must hold down the key labeled Ctrl while you press another key or a pointing device button.
<b>KPn</b>	A sequence such as KP1 indicates that you must first press and release the key labeled KP1, then press and release another key or a pointing device button.
...	In examples, a horizontal ellipsis indicates one of the following: <ul style="list-style-type: none"> <li>• Additional optional arguments in a statement have been omitted.</li> <li>• The preceding item or items can be repeated one or more times.</li> <li>• Additional parameters, values, or other information can be entered.</li> </ul>
.	A vertical ellipsis indicates the omission of items from a code example or command format; the items are omitted because they are not important to the topic being discussed.
( )	In format descriptions, parentheses indicate that, if you choose more than one option, you must enclose the choices in parentheses.

<b>Conventions</b>	<b>Description</b>
[ ]	In format descriptions, brackets indicate that whatever is enclosed is optional; you can select none, one, or all of the choices.
{ }	In format descriptions, braces surround a required choice of options; you must choose one of the options listed.
User Input	The hardcopy version of this manual has interactive examples that show user input in red letters and system responses or prompts in black letters. The online version differentiates user input from system responses or prompts by using a different font.
<b>boldface text</b>	Boldface words introduce new terms that are defined in the glossary.
<i>italic text</i>	Italicized words introduce new terms.
UPPERCASE TEXT	Uppercase letters indicate the name of a command or routine. Lowercase words and letters used in examples indicate that you should substitute a word or value of your choice.
mouse	The term <i>mouse</i> is used to refer to any pointing device, such as a mouse, a puck, or a stylus.
MB1, MB2, MB3	MB1 indicates the left mouse button, MB2 indicates the middle mouse button, and MB3 indicates the right mouse button. (The buttons can be redefined by the user.)

# Introduction to DEC/Test Manager

---

This chapter describes the DEC/Test Manager, environment, and components. A sample session provides an overview of DEC/Test Manager, and a section on entering commands.

---

## 1.1 Overview

DEC/Test Manager is a software development and maintenance tool that organizes and automates the software regression testing process. You use DEC/Test Manager to run, review, and store software regression tests and test results.

**Regression testing** ensures that an application runs correctly and that new features you add to an application do not affect the correct execution of previously tested features. If errors do occur, the application has regressed.

The following list outlines typical regression testing steps for an application:

1. Create tests for the application.
  - a. Organize the tests.
  - b. Create a mechanism to allow ready access to tests.
2. Run the tests.
3. Examine the test results.
  - a. Compare the test results to those you expected and note any differences.
  - b. Revise the application code to correct problems that caused incorrect test output. Repeat steps 2 and 3 until the test output is correct.
4. Save the correct output as the validated test results.

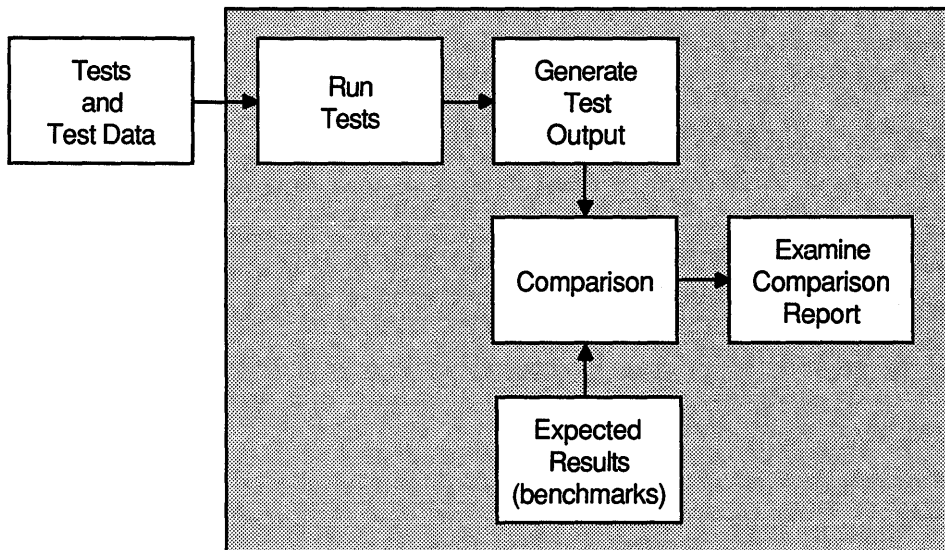
5. Repeat steps 2 through 4 whenever you modify the application or add new code.

If the current and the previously validated test results match, the application being tested is working as expected.

If you find unexpected changes in test results, the application being tested may contain errors.

DEC/Test Manager automates the regression testing steps except for the creation of tests, which only you can do for your software applications. Figure 1-1 shows the regression testing steps. The shaded area indicates those steps that DEC/Test Manager automates.

**Figure 1-1: Regression Testing with DEC/Test Manager**



ZK-2084-GE

DEC/Test Manager performs the following actions with minimal or no user assistance:

- Organizes test files.
- Runs tests.



- Compares the current results with the expected results in the **benchmark file** and logs any differences found between benchmark and current results. (A benchmark file contains the expected output for the test's execution.)
- Saves the test results if different from the benchmark.
- Displays the test results for examination.

You can use DEC/Test Manager to test software that executes in a variety of common VMS environments. Table 1–1 shows the environments that DEC/Test Manager supports.

**Table 1–1: Supported DEC/Test Manager Environments**

<b>Chosen Environment</b>	<b>Result</b>
Noninteractive	You can use DEC/Test Manager with noninteractive tests that do not have a terminal-oriented or workstation interface. Software that accepts an input text file and gives you an output text file can be tested as a noninteractive DEC/Test Manager test.
Interactive Terminal	You can use DEC/Test Manager with interactive terminal tests for testing of software with an interactive terminal-oriented interface.
Interactive DECwindows	You can use the DEC/Test Manager with DECwindows tests for testing of software that presents a windowed interface within the DECwindows workstation environment.

Table 1–2 further describes the DEC/Test Manager environment by describing DEC/Test Manager terms.

**Table 1–2: DEC/Test Manager Terms**

<b>Term</b>	<b>Description</b>
Library	DEC/Test Manager stores the information it needs to manage a test system in a VMS directory called a <b>DEC/Test Manager library</b> . Chapter 3 describes DEC/Test Manager libraries in detail.

(continued on next page)

**Table 1–2 (Cont.): DEC/Test Manager Terms**

<b>Term</b>	<b>Description</b>
Test Description	A collection of fields that contain the information DEC/Test Manager needs to run a particular test. A test description requires a template file and can have other optionally specified test-related entities. A <b>template file</b> is a VMS command procedure that executes a noninteractive test, or a session file containing a recorded interactive terminal or DECwindows session. Chapter 3 describes tests and test descriptions in more detail.
Collection	A set of tests selected for execution. You can execute a test only in the context of a collection. You can select tests for inclusion in a collection by test name or groups. Chapter 4 describes collections in detail.

## 1.2 Entering Commands

You can enter DEC/Test Manager commands in several ways:

- From the Digital Command Language (DCL) command line
- From the DEC/Test Manager command line
- From the DECwindows interface (see Chapter 2)
- From a program that calls DEC/Test Manager directly (see Chapter 10)

When you enter a command from the DEC/Test Manager command line, you omit the DTM command. After the command executes, control returns to the DEC/Test Manager subsystem level.

The following example shows how to invoke DEC/Test Manager, issue the SHOW VERSION command, and exit from DEC/Test Manager:

```
$ DTM
DTM> SHOW VERSION
DEC/Test Manager Version 3.1
DTM> EXIT
$
```

If you press RETURN before completing a command, you are prompted for all required information for a command.

If you plan to enter many commands, you should use DEC/Test Manager as a subsystem to avoid the processing overhead that occurs when you invoke DEC/Test Manager from DCL directly.

## NOTE

Examples in this manual show prompts from the following process levels:

- The dollar sign prompt ( \$ ) indicates DCL level.
- The DEC/Test Manager prompt (DTM>) means a command is being issued from the DEC/Test Manager subsystem; examples in this manual typically show commands entered from this prompt.
- The DTM\_REVIEW> prompt means a command is being issued from the Review subsystem of DEC/Test Manager.

---

### 1.2.1 Getting Help

You can access DEC/Test Manager online help in several ways. To access the DEC/Test Manager help system, type the following command at the DCL prompt:

```
$ HELP DTM
```

To access help on a specific command, type the command name at the DCL prompt:

```
$ HELP DTM COPY TEST_DESCRIPTION
```

You can also access DEC/Test Manager help from the DEC/Test Manager command line or any of the DEC/Test Manager subsystem command lines.

```
$ DTM  
DTM> HELP COPY TEST_DESCRIPTION
```

---

### 1.2.2 Canceling Commands

If you want to cancel a command before it has completed, press CTRL/C. If you press CTRL/C during a wildcard transaction that updates the library, DEC/Test Manager completes the current transaction, but does not continue.

When you enter a DEC/Test Manager command from DCL and then press CTRL/C during execution of the command, DEC/Test Manager returns control to DCL level. If you enter the command from the DEC/Test Manager subsystem prompt level, DEC/Test Manager obtains control as indicated by the DTM> prompt.

---

## 1.3 Getting Started

This section shows fundamental DEC/Test Manager features. To get you started using DEC/Test Manager, this section shows a DEC/Test Manager session with an interactive terminal test. Noninteractive and DECwindows tests are described later in this manual.

The example in this section is designed so that you can recreate a DEC/Test Manager session at a terminal or workstation. The interactive terminal example demonstrates the following:

- Invoking DEC/Test Manager
- Creating a new DEC/Test Manager library
- Selecting an existing DEC/Test Manager library
- Creating a test
- Showing a test within DEC/Test Manager
- Recording a test
- Creating a collection
- Showing a collection within DEC/Test Manager
- Executing the test in a collection
- Comparing the test results to the expected output
- Examining the test results

Example 1-1 uses the VMS Mail Utility (MAIL) and shows you how to use DEC/Test Manager to test some of the MAIL commands.

The reverse numbers refer you to the command line explanation in the list that follows the example.

Phrases enclosed in quotation marks ( " " ) are remarks that are associated with the command being issued. For most commands, DEC/Test Manager prompts you for a remark if you do not include one on the command line. A null remark string is permitted.

## Example 1-1: Sample Interactive Terminal Session

---

```
① $ CREATE/DIRECTORY [.DTMLIB]
② $ DTM
③ DTM> CREATE LIBRARY [.DTMLIB] "New DEC/Test Manager Library"
%DTM-S-CREATED, DEC/Test Manager library DUA1:[USER01.DTMLIB] created
④ DTM> CREATE TEST_DESCRIPTION MAIL_TEST/INTERACTIVE
⑤ Remark: Going to record a MAIL test
%DTM-I-DEFAULTED, benchmark file name defaulted to MAIL_TEST.BMK
%DTM-I-DEFAULTED, template file name defaulted to MAIL_TEST.SESSIO
%DTM-S-CREATED, test description MAIL_TEST created
⑥ DTM> SHOW TEST_DESCRIPTION

Test Descriptions in DEC/Test Manager Library DUA1:[USER01.DTMLIB]

MAIL_TEST      "Going to record a MAIL test"
  Template      = MAIL_TEST.SESSIO
  Benchmark     = MAIL_TEST.BMK
  Prologue     = None Specified
  Epilogue     = None Specified
⑦ DTM> RECORD MAIL_TEST "Recording Mail on the terminal"
%DTM-I-BEGIN, your interactive test session is now beginning...
Type CTRL/P twice to terminate the session.

⑧ $ SET BROADCAST=NONE
⑨ $ MAIL
⑩ MAIL> SHOW PERSONAL_NAME
Your personal name is "DEC/Test Manager - Project Q41327".
⑪ MAIL> SET PERSONAL_NAME "DEC/Test Manager - Engineer USER01"
⑫ MAIL> SHOW personal_name
Your personal name is "DEC/Test Manager - Engineer USER01".
⑬ MAIL> EXIT
⑭ $ set broadcast=all
⑮ $ ^P ^P

^P

%DTM-I-BMK_SAVED, benchmark has been saved in file DUA1:[USER01.DTMLIB]MAIL_TEST
.BMK;1
%DTM-S-RECORDED, test MAIL_TEST has been successfully recorded in file DUA1:[USE
R01]MAIL_TEST.SESSIO
```

---

(continued on next page)

## Example 1-1 (Cont.): Sample Interactive Terminal Session

---

```
16 DTM> CREATE COLLECTION MAIL_COLL MAIL_TEST "Creating the MAIL test collection"
%DTM-S-CREATED, collection MAIL_COLL created

17 DTM> SHOW COLLECTION

Collections in DEC/Test Manager Library DUAL:[USER01.DTMLIB]

MAIL_COLL      1 test      27-OCT-1989 09:49:38
      Command: CREATE COLLECTION MAIL_COLL MAIL_TEST "Creating the MAIL test
              collection"
      Status: not run

18 DTM> RUN MAIL_COLL

Starting MAIL_TEST test run...

%DTM-I-BEGIN, your interactive test session is now beginning...
$ SET BROADCAST=NONE
$ MAIL

19 MAIL> SHOW PERSONAL_NAME
Your personal name is "DEC/Test Manager - Engineer USER01".

MAIL> SET PERSONAL_NAME "DEC/Test Manager - Engineer USER01"

MAIL> SHOW PERSONAL_NAME
Your personal name is "DEC/Test Manager - Engineer USER01".

MAIL> EXIT
$ SET BROADCAST=ALL
$
%DTM-S-CONCLUDED, your interactive test session has concluded

Performing post-run cleanup with comparison...

%DTM-I-UNSUCCESS, the comparison for the test MAIL_TEST was unsuccessful
%DTM-S-COMPARED, collection MAIL_COLL compared

20 DTM> REVIEW MAIL_COLL
Collection MAIL_COLL with 1 test was created on 27-OCT-1989 09:49:38 by the
command:
      CREATE COLLECTION MAIL_COLL MAIL_TEST "Creating the MAIL test
collection"
      Last Review Status = not previously reviewed
      Success count = 0
      Unsuccessful count = 1
      New test count = 0
      Updated test count = 0
      Comparisons aborted = 0
      Test not run count = 0

21
      Result Description MAIL_TEST      Comparison Status : Unsuccessful
```

---

(continued on next page)

## Example 1-1 (Cont.): Sample Interactive Terminal Session

---

```
DTM_REVIEW> SHOW/DIFFERENCES
.
.
DTM_REVIEW> EXIT
%DTM-S-EXIT, leaving Review subsystem
DTM> EXIT
$
```

---

- ① Create an empty subdirectory for use as a DEC/Test Manager library.
- ② Invoke DEC/Test Manager.
- ③ Create a new DEC/Test Manager library and assign it to the empty subdirectory. When you create a library, DEC/Test Manager automatically sets this as the current library; you do not need to explicitly set the library.
- ④ Create the test description called MAIL\_TEST and designate it as an interactive terminal test.
- ⑤ When you do not supply a comment to the CREATE TEST\_DESCRIPTION command, DEC/Test Manager prompts you for one.
- ⑥ Show the test descriptions in the current library.
- ⑦ Record the MAIL\_TEST test. Note that DEC/Test Manager spawned to DCL.
- ⑧ Disallow messages to be broadcast on the terminal so that incoming messages will not interfere with the test recording.
- ⑨ Invoke the Mail Utility.
- ⑩ Show the current personal name. In this case, no personal name was set.
- ⑪ Set the personal name.
- ⑫ Show the personal name again.
- ⑬ Exit from the Mail Utility.
- ⑭ Reset the terminal to receive broadcast messages.
- ⑮ End the recording session by pressing CTRL/P twice. The ^P characters are not echoed on the terminal, but a ^P character appears below the place where you ended the recording session.
- ⑯ Create the collection MAIL\_COLL and include the MAIL\_TEST test in it.

- 17 Show the collection information.
- 18 Run the MAIL\_COLL collection interactively.
- 19 Note here that when DEC/Test Manager issues the SHOW PERSONAL NAME command from the session file, the Mail Utility displays the previously stored personal name and not the message that was displayed when the test was recorded. The result of this command will cause the test to be unsuccessful.
- 20 After the collection run ends and the test has been compared, invoke the Review subsystem to review the MAIL\_COLL collection. DEC/Test Manager automatically displays collection statistics for MAIL\_COLL.
- 21 DEC/Test Manager marked the test as unsuccessful because the result file differed from the benchmark file.
- 22 Enter the SHOW/DIFFERENCES command. The screens were omitted from this example but you can display them by pressing the RETURN key or specifying the NEXT command to view subsequent difference file records.
- 23 Type CTRL/Z to exit from the Review subsystem.
- 24 Enter the EXIT command to exit from the Review subsystem.
- 25 Enter the EXIT command to exit from DEC/Test Manager.



# Using DEC/Test Manager in DECwindows

---

This chapter describes using DEC/Test Manager in a DECwindows environment; it provides an overview and a sample session.

---

## 2.1 Overview

To invoke the DEC/Test Manager DECwindows interface, enter the following command from a DECwindows terminal emulator window:

```
$ DTM/INTERFACE=DECWINDOWS
```

Figure 2-1 shows the initial DEC/Test Manager title bar and the main menus you can select from the menu bar.

Figure 2–1: DEC/Test Manager DECwindows Title Bar and Main Menus

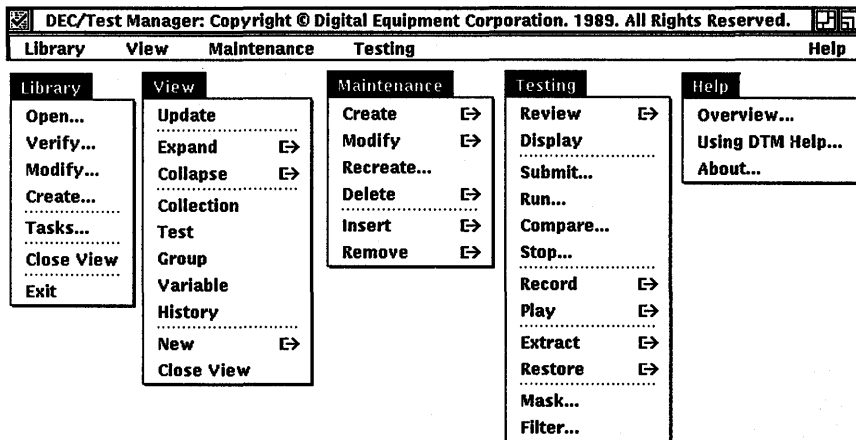


Figure 2–1 shows the menus separated from the menu bar to show you all the DEC/Test Manager main menus at once; you can pull down only one main menu at a time.

## 2.1.1 Getting Help

You can obtain DEC/Test Manager Help in a DECwindows environment by pulling down the Help menu.

You can also get context-sensitive help in the following way:

1. Press and hold the HELP key (F15).
2. Move the pointer to the item you want help on and release MB1.

## 2.1.2 Displaying DEC/Test Manager Information in DECwindows

You can display files and view information on result descriptions, collection tests, groups, variables, and history through **views**. The views are the DECwindows equivalent of the character cell interface SHOW commands.

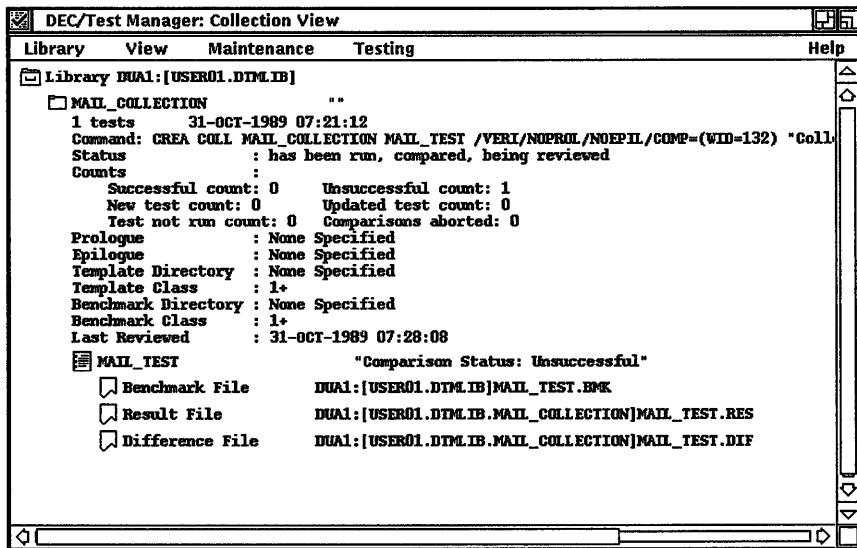
DEC/Test Manager displays the collection view when you invoke DEC/Test Manager. To obtain a view of collections, tests, groups, variables, or history, perform the following steps:

1. Pull down the View menu.
2. Choose one of the views from the menu.

You can obtain more detail by double clicking on an item (test, group, collection, variable, or library). If an item is expanded fully, double clicking collapses the item into the previous level of information.

Figure 2-2 shows a collection view with the MAIL\_COLLECTION collection expanded.

**Figure 2-2: Expanded Collection View**



To view result, benchmark, or difference files, expand the view as described in the previous steps and double click on a file name.

---

### 2.1.3 DEC/Test Manager Command Correlation

Most DEC/Test Manager commands have a corresponding menu path in the DECwindows interface; however, there is not a complete one-to-one correspondence because of added functions available in the DECwindows interface.

There is no corresponding DECwindows action for the DEC/Test Manager SPAWN and ATTACH commands, because DECwindows enables you to create another process without having to spawn out of a process.

---

## 2.2 Sample DECwindows Session

This sample session uses the DECwindows Mail Utility (MAIL) to show you how to use DEC/Test Manager to test a DECwindows application. This section describes the following DEC/Test Manager DECwindows topics:

- Creating a DEC/Test Manager library
- Creating a test description
- Recording a test
- Creating a collection
- Running a collection
- Reviewing the results
- Updating the benchmark
- Creating masked regions

#### NOTE

Many of the figures in this section show dialog boxes from which you initiate tasks. These figures also show the menu and menu item from which the dialog box is invoked.

---

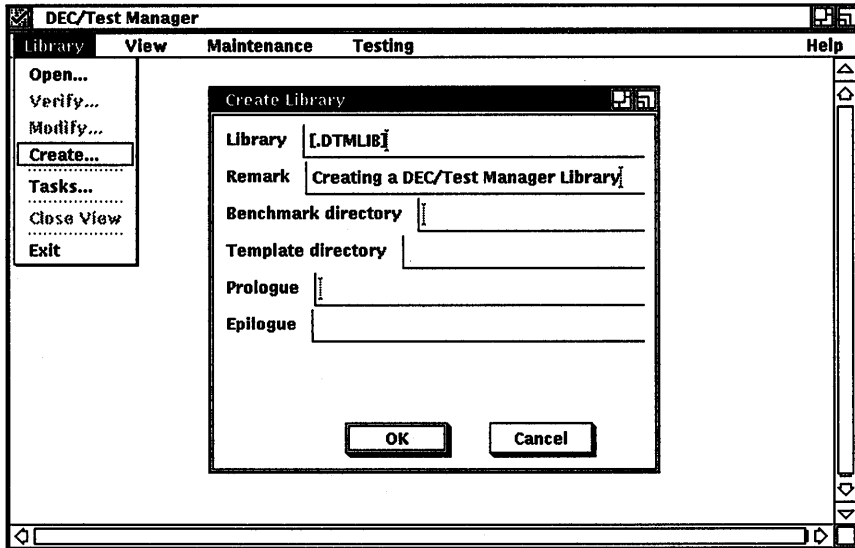
### 2.2.1 Creating a Library

Figure 2-3 shows how to create a new DEC/Test Manager library. If you want to specify an existing library, choose the Open... menu item and specify the library in the subsequent dialog box.

## NOTE

You must create an empty VMS directory before you can create a DEC/Test Manager library.

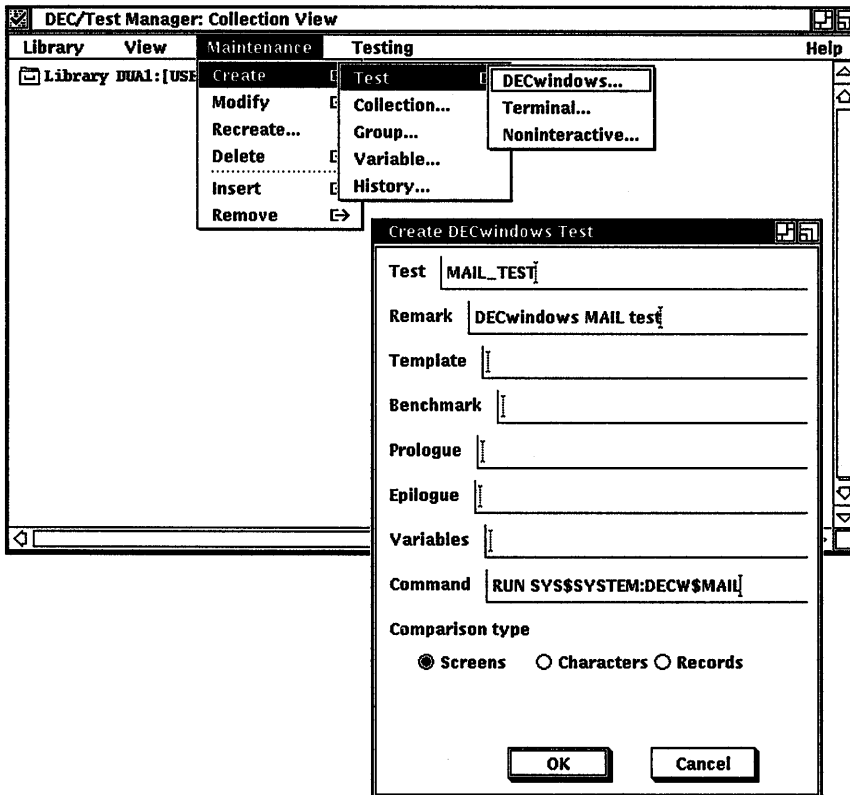
**Figure 2-3: Creating a DEC/Test Manager Library**



### 2.2.2 Creating a Test

Figure 2-4 shows how to create a DECwindows test. The command to run the DECwindows Mail facility is on this dialog box.

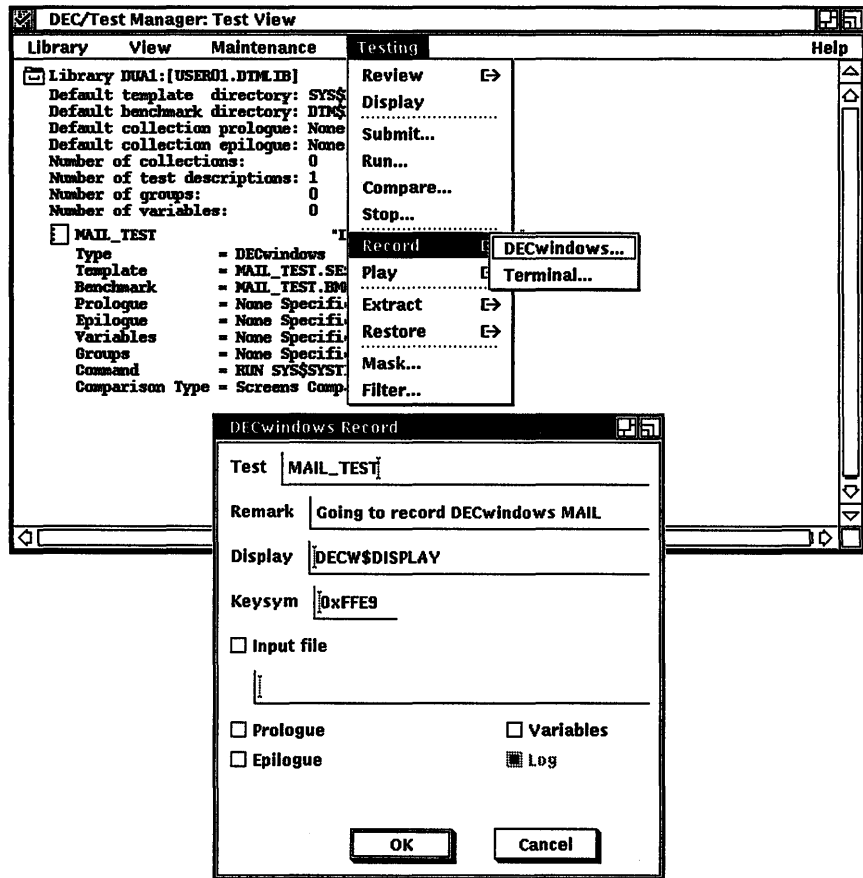
Figure 2-4: Creating a DECwindows Test



### 2.2.3 Recording a Test

Figure 2-5 shows a sample Record dialog box and how to invoke it. If you click on the test name before invoking the Record dialog box, the test name automatically appears in the Test field.

Figure 2-5: Recording a DECwindows Test



When you record the DECwindows test, ensure that the conditions of the test at its start are the same as at its end. This enables you to run a DECwindows input file without dependencies on an initialized workstation.

For example, creating a solid background, making icons of all applications, placing the icons in a consistent order all help to ensure that start and end conditions are equal. See Section 3.5.3 for more information about recording DECwindows tests.

Figure 2-6 shows the Ready to Record dialog box. After you acknowledge the Ready to Record dialog box, you can set the conditions of the test before you press the Compose Character key and the S key at the same time to start recording the test.

Figure 2-6: Ready to Record Dialog Box

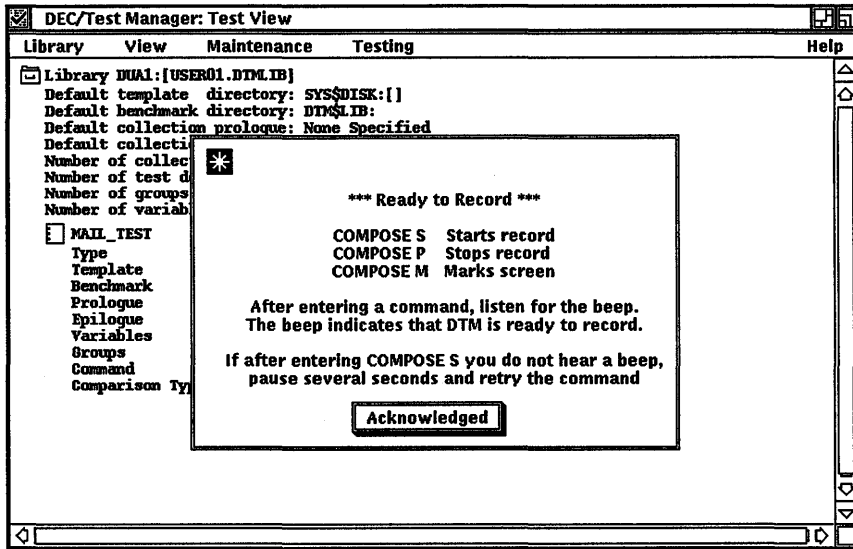
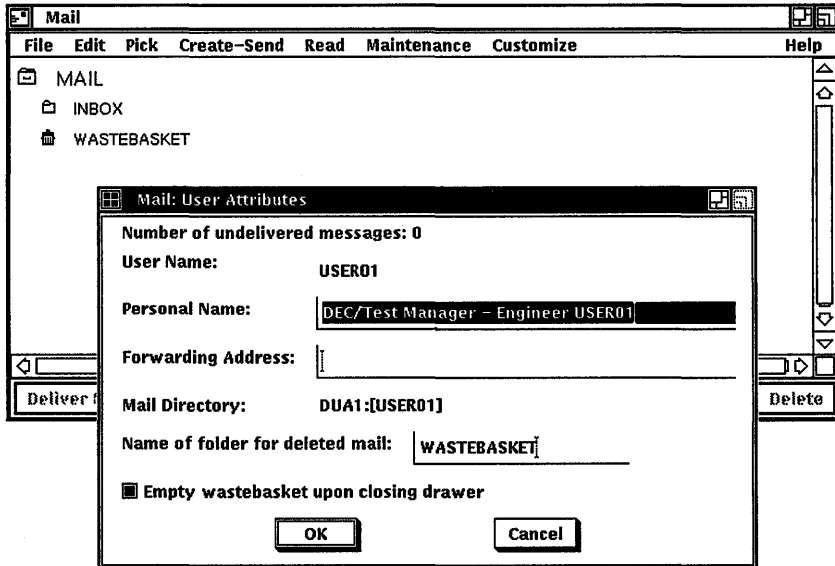




Figure 2-7 shows the DECwindows Mail Utility window and the User Attributes dialog box.

**Figure 2-7: Sample DECwindows Recording Session**



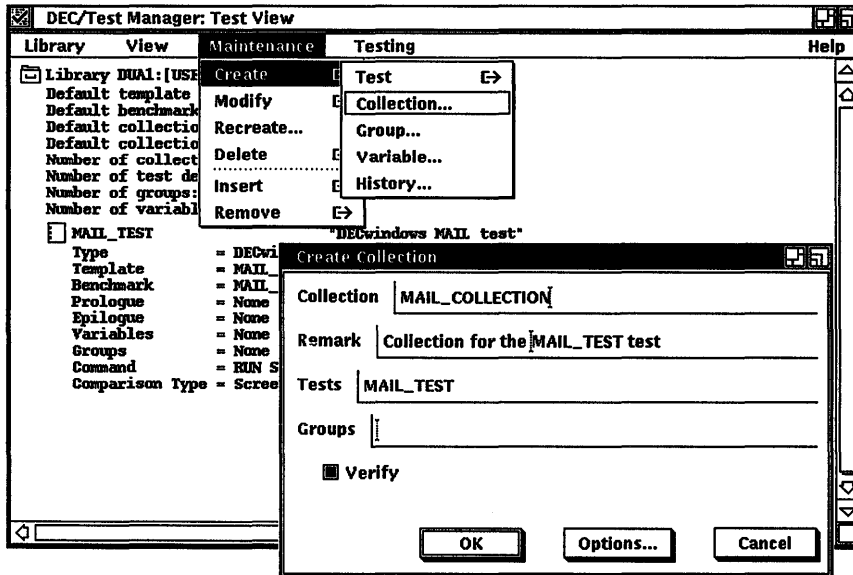
When Figure 2-7 is displayed on your screen, you press the Compose Character key and the M key at the same time; this marks the screen for comparison in subsequent test executions. Because you only test the attributes in the User Attributes dialog box, you need mark only this screen. See Chapter 3 for more information on recording sequence keys.

When you finish recording the test, press the Compose Character key and the P key at the same time.

## 2.2.4 Creating a Collection

Figure 2–8 shows how to create the MAIL\_COLLECTION collection. The MAIL\_TEST test is the only test in the collection.

Figure 2–8: Creating a Collection



### NOTE

Before you execute the MAIL\_COLLECTION collection, you must modify the attributes on the User Attributes dialog box to simulate a change in the MAIL application so that differences occur in the comparison; this is done to emulate changes in software. In the sample session, the User Attributes dialog box is modified in the following ways:

- The personal name is changed from “DEC/Test Manager - Engineer USER01” to “DEC/Test Manager - Project Q121459”.
- A forwarding address is added: USER44.
- The “WASTEBASKET” folder specification is changed to “WASTE”.

- The Empty Wastebasket Upon Closing Drawer button is disabled.

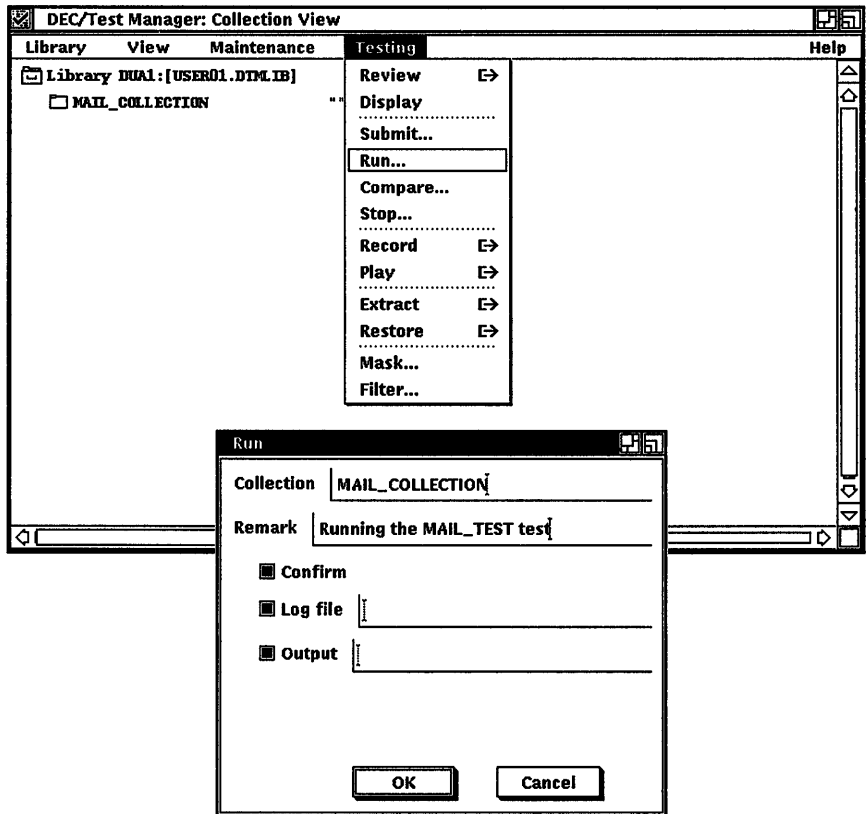
---

## 2.2.5 Executing a Collection

Figure 2-9 shows how to execute the MAIL\_COLLECTION collection. After specifying the Collection and Remark fields, click on the OK button.

**Figure 2-9: Executing a Collection**

---



---

## 2.2.6 Displaying a Test Result

When you review a DECwindows test, DEC/Test Manager provides you with a benchmark image file and a result image file, and shows any differences between the two.

DEC/Test Manager displays screen images in windows that you can **scroll** and **pan** with resize and reposition capabilities. Scrolling and panning provide screen image movement within a window.

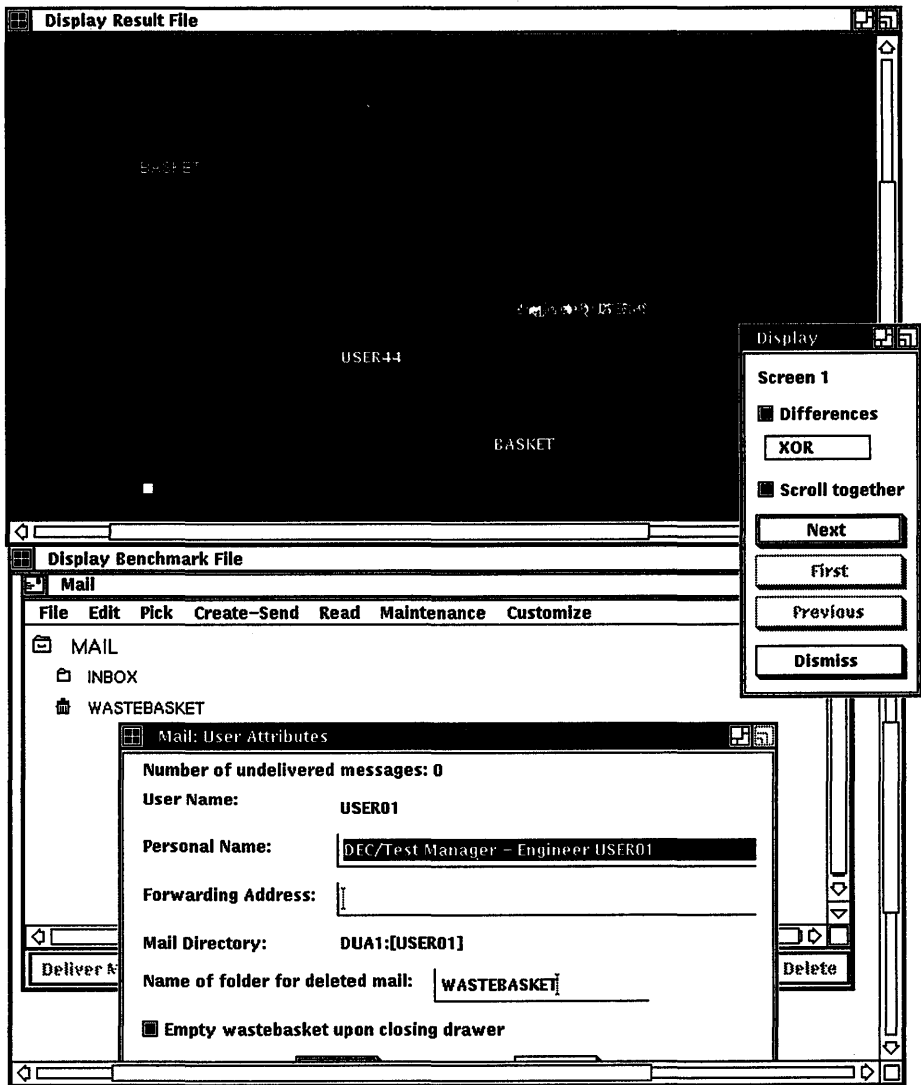
To view the differences in the MAIL\_TEST benchmark and result files, expand the collection and test to show the benchmark, result, and difference files. Then double click on the difference file.

Figure 2–10 shows the differences between the MAIL\_TEST benchmark and the MAIL\_TEST result files. The differences are shown in XOR (exclusive OR) mode, which means that when the pixels are compared, those that do not match are displayed (Figure 2–10 shows these in white); those that match are not displayed.

The following list describes the differences:

- The change from “WASTEBASKET” to “WASTE” shows the difference (“BASKET”) in two places.
- The addition of the forwarding address, USER44, shows with the text entry cursor also moved.
- The disabled button is visible.
- The jumbled part of the difference image is where the characters “Engineer USER01” and “Project Q121459” are superimposed on one another; DEC/Test Manager shows the pixels that do not match between these two sets of characters.

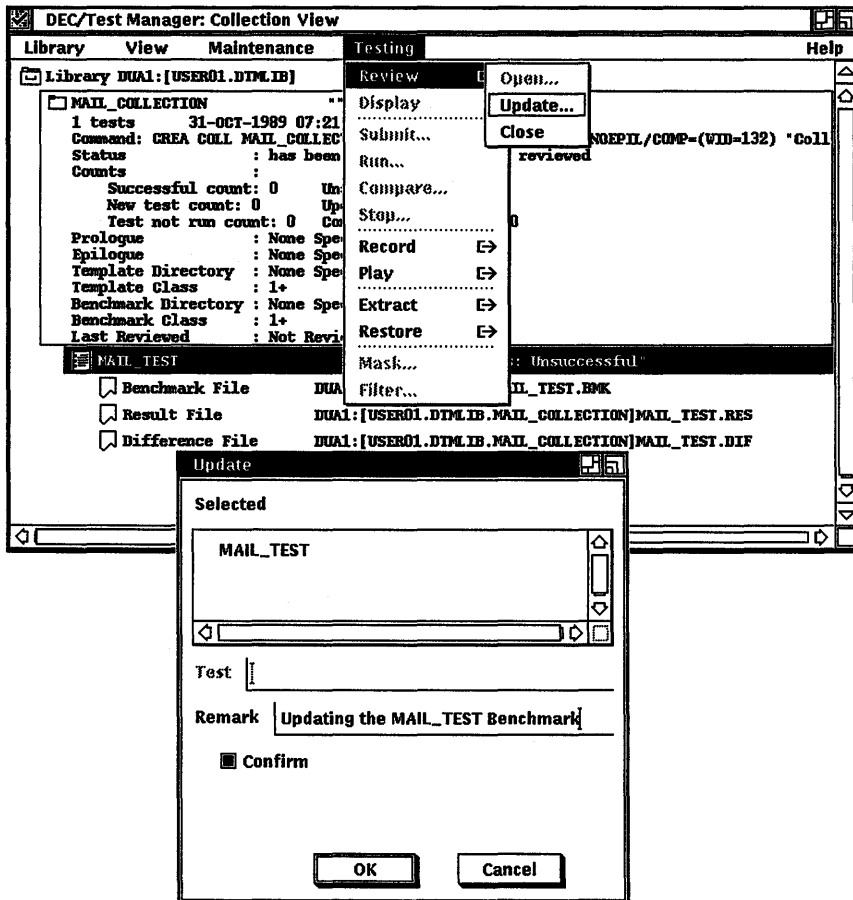
Figure 2-10: Viewing Differences



## 2.2.7 Updating a Benchmark File

Figure 2-11 shows you how to update the benchmark file for the MAIL\_TEST test. Note that the test was selected before choosing the Update menu item.

Figure 2-11: Updating a Benchmark Image



---

## 2.2.8 Creating a Benchmark Mask

Several factors can cause a test to fail with undesirable results. For example, if mail is received during the sample session described in this chapter, the icons in the DECwindows Mail facility can become altered. To ensure that areas of a DECwindows application that you have no interest in do not affect a test result comparison, you can create areas called **masks** on benchmark images using the DEC/Test Manager Mask Editor. Areas that are masked are not compared when DEC/Test Manager compares the results of a test execution against the benchmark image.

Generally, areas are masked on a benchmark image after recording the test and before executing the test to mask out run-dependent image data and maximize the chances for successful comparison status for the test. However, you can create masked areas on a benchmark image at any time.

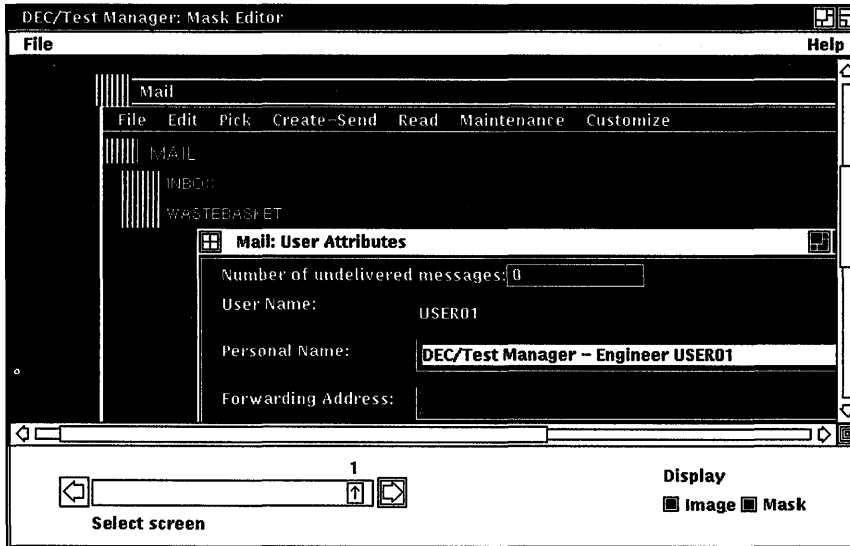
To invoke the Mask Editor, click on the DECwindows test then pull down the Testing menu and choose the Mask... menu item. The benchmark image is automatically loaded into the Mask Editor.

To create a mask on a benchmark image, perform the following steps:

1. Move the pointer to the beginning of the area to mask and press and hold MB1.
2. Move the pointer to the opposite corner of the area to mask and release MB1.

Figure 2-12 shows the benchmark image from Figure 2-7 with three masks already defined over some icons (shown as striped rectangles) and a mask in the process of being defined over the Number of Delivered Messages number field (when MB1 is released, it will also display a striped rectangle).

**Figure 2-12: Applying Masks to a Benchmark Image**



To save a masked image file, pull down the File menu and choose the Save menu item; the Mask Editor saves the image and masks in the same file name as the one you read into the Mask Editor.

Pull down the File menu and choose Quit before saving the image file and masks to leave the Mask Editor without updating the image file; choose Quit after saving the image file and masks to exit from the Mask Editor.

You can delete a mask by double clicking on a defined mask.

You can move a mask by placing the pointer on the mask you want to move, pressing and holding MB2, and releasing MB2 when you move the mask to the new area.

Masked areas do not become part of the benchmark image. DEC/Test Manager stores the coordinates of the masked areas in the image file, but you determine whether to compare the image with or without the mask. If you create masked areas on a benchmark file, the masks are used by default. If you do not want previously-created mask areas to be applied to a result comparison, you must explicitly specify ignoring masks on the Create Collection or Compare dialog boxes.



The three basic components of a DEC/Test Manager test system are as follows:

- The DEC/Test Manager library
- Tests
- Collections

This chapter describes libraries and tests and provides information on the following topics:

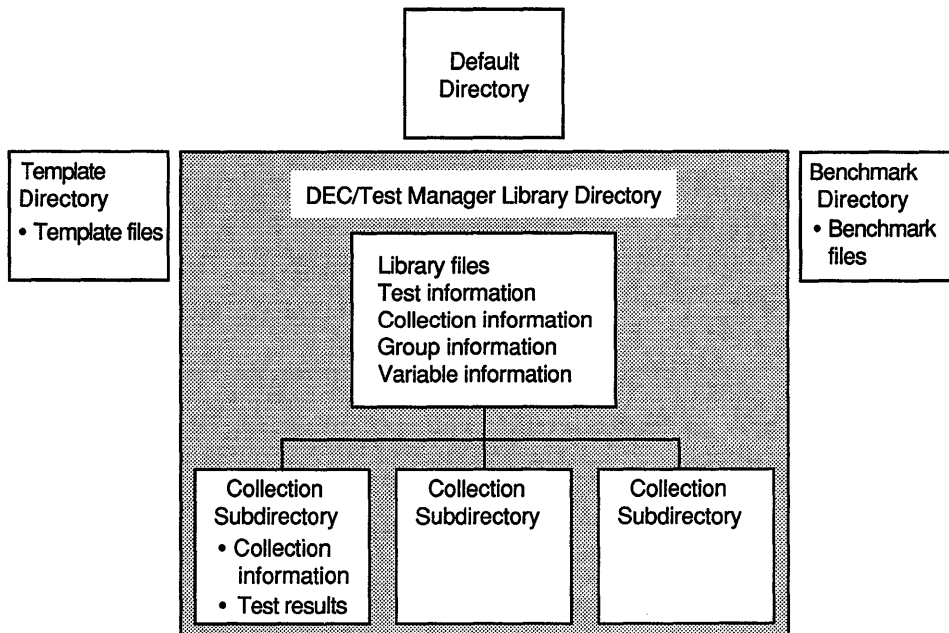
- DEC/Test Manager history
- Test descriptions
- Noninteractive tests
- Interactive terminal tests
- Input files created from session files

---

### 3.1 DEC/Test Manager Libraries

DEC/Test Manager stores the information it needs to manage a test system in a VMS directory called a DEC/Test Manager library. Figure 3–1 shows a customized DEC/Test Manager library and its structure, showing template files and benchmark files stored in their own directories instead of in the default directory and the DEC/Test Manager library directory.

**Figure 3-1: Overview of a Custom DEC/Test Manager Library**



ZK-2082-GE

### 3.1.1 Creating a DEC/Test Manager Library

To create a DEC/Test Manager library, you must first create a VMS directory, then invoke DEC/Test Manager and enter the **CREATE LIBRARY** command as shown in the following procedure:

```
$ CREATE/DIRECTORY [.DTMLIB]
$ DTM
DTM> CREATE LIBRARY [.DTMLIB] "New DEC/Test Manager library"
%DTM-S-CREATED, DEC/Test Manager library DUA0:[USER01.DTMLIB] created
DTM>
```

The phrase enclosed in quotation marks ( " ") following the library specification is a remark that you associate with the library you are creating. DEC/Test Manager prompts you for a remark if you do not include one on the command line, but a null remark string is permitted.

DEC/Test Manager creates a subdirectory in the library for each collection you create. Do not create subdirectories or files in a directory containing the DEC/Test Manager library or any of its subdirectories. Do not set the default directory to be a DEC/Test Manager library or any of its subdirectories.

If you put files in these subdirectories, they are deleted along with the collection files when you instruct DEC/Test Manager to delete the collections.

#### NOTE

VMS limits directory trees to a depth of eight; because DEC/Test Manager may create subdirectories, you should not create a library in an eighth-level directory.

---

### 3.1.2 Setting a Library

When you invoke DEC/Test Manager, you must explicitly specify the library you want to use to store files. You do this by selecting an existing library by using the SET LIBRARY command.

The CREATE LIBRARY command performs an implicit SET LIBRARY command so that you can use DEC/Test Manager commands with the library just created. However, after you have created one or more DEC/Test Manager libraries, you must set the default DEC/Test Manager library for subsequent sessions. For example, to select the library you created in Section 3.1.1, type the following command at the DCL prompt:

```
$ DTM SET LIBRARY [.DTMLIB]
%DTM-S-LIBIS, DEC/Test Manager library is DUA0:[USER01.DTMLIB]
```

After you select a library, all DEC/Test Manager commands you enter refer to that library until you select another library, create another library, or log out.

#### Setting a Benchmark Directory

DEC/Test Manager places benchmark files in the current DEC/Test Manager library. You can specify another directory using the SET BENCHMARK\_DIRECTORY command, as follows:

```
DTM> SET BENCHMARK_DIRECTORY DUA0:[USER01.BMK]
_Remark: "New default benchmark directory"
%DTM-S-NEWDEF, DUA0:[USER01.BMK] is the new default collection benchmark
directory
DTM>
```

See Chapter 7 for information about storing files outside the DEC/Test Manager library, such as in a VAX DEC/Code Management System (CMS) library.

## Setting a Template Directory

DEC/Test Manager places template files in the current directory. You can specify another directory using the `SET TEMPLATE_DIRECTORY` command, as follows:

```
DTM> SET TEMPLATE_DIRECTORY DUA0:[USER01.TMPL]
_Remark: "New default template directory"
%DTM-S-NEWDEF, DUA0:[USER01.TMPL] is the new default
collection template directory
DTM>
```

See Chapter 7 for information about storing files outside the DEC/Test Manager library.

### NOTE

In a DECwindows environment, you can set default benchmark and template directories and collection prologue and epilogue files using the Create Library or Modify Library dialog box.

---

## 3.1.3 Displaying DEC/Test Manager Library Information

You can obtain library information in two forms:

- As a directory and file specification
- As a summary

Use the `SHOW LIBRARY` command to display library directory and file specification information for the current library. For example:

```
DTM> SHOW LIBRARY
Your DEC/Test Manager library is DUA0:[USER01.DTMLIB]
```

Use the `SHOW ALL` command to display library summary information for the current library. For example:

```
DTM> SHOW ALL
```

```
Description of DEC/Test Manager Library DUA0:[USER01.DTMLIB]
```

```
Default template directory: DUA0:[USER01.TEMPLATES] ""
Default benchmark directory: DUA0:[USER01.BENCHMARKS] ""
Default collection prologue: None Specified
Default collection epilogue: None Specified
Number of collections:      20
Number of test descriptions: 152
Number of groups:          18
Number of variables:       9
```

If one of these entities does not exist for the library, the message NONE SPECIFIED is displayed. You can also place the library summary information into a file by specifying the /OUTPUT qualifier. See the Command Dictionary for more information about the SHOW LIBRARY command.

---

### 3.1.4 DEC/Test Manager History

When you create a new DEC/Test Manager library, DEC/Test Manager automatically creates a history for that library. Whenever you issue a DEC/Test Manager command that alters the library, DEC/Test Manager enters that command and its associated remark into the history.

The SHOW HISTORY command displays a chronological list of library transactions. See the SHOW HISTORY command in the Command Dictionary for a list of the commands that are logged into the history.

You can list all of the history transactions by entering the following command:

```
DTM> SHOW HISTORY
History in DEC/ TEST MANAGER Library DUA0:[USER01.DTMLIB]
25-JAN-1990 12:03:54 USER01 CREATE LIBRARY DUA0:[USER01.DTMLIB] "Test
library"
25-JAN-1990 12:04:12 USER01 CREATE TEST_DESCRIPTION MAIL_TEST/INTERACTIVE
"Going to record a MAIL test"
25-JAN-1990 12:05:32 USER01 RECORD MAIL_TEST "Recording MAIL on the terminal"
.
.
.
DTM>
```

You can also choose the types of history transactions that you want to display. The following example instructs DEC/Test Manager to display all the RECORD commands for the MAIL\_TEST tests that were made by user USER01.

```
DTM> SHOW HISTORY MAIL_TEST/TRANSACTION=RECORD/USER=USER01
History in DEC/TEST MANAGER Library DUA0:[USER01.DTMLIB]
25-JAN-1990 12:05:32 USER01 RECORD MAIL_TEST "Recording MAIL on the terminal"
.
.
.
DTM>
```

---

### 3.1.4.1 Adding a Remark to the History

Use the **REMARK** command to add a remark to the history; for example, to note an unusual occurrence or mark a milestone in the testing system. The following example shows a remark being added to the history:

```
DTM> REMARK "End of Version 3.0 Testing"
%DTM-S-REMARK, remark added to history file
DTM>
```

DEC/Test Manager enters the remark into the history, as in the following example:

```
14-NOV-1990 13:01:32 USER01 REMARK "End of Version 3.0 Testing"
```

---

### 3.1.4.2 Deleting History Information

You can delete all or part of the history information for a DEC/Test Manager library with the **DELETE HISTORY** command. Deleted history information is written to a **HISTORY.OUT** file in the default directory. Once history information is deleted from the history, it cannot be replaced. DEC/Test Manager enters the deletion in the history as in the following example:

```
1-JUN-1990 15:03:55 USER01 REMARK "Deleting the old information"
```

You can delete the entire history information, as shown in the following example. DEC/Test Manager prompts you with the current date and time from which to delete the history records back to the first record.

```
DTM> DELETE HISTORY "Deleting all the old information"
Confirm DELETE HISTORY/BEFORE=14-Nov-1990 [Y/N] (N): Y
%DTM-S-HISTDEL, 323 history records deleted
DTM>
```

Also, you can delete a portion of the history information from a specified date and time back to the beginning of the history, as shown in the following example:

```
DTM> DELETE HISTORY/BEFORE=1-JUN-1990 "Deleting the old information"
Confirm DELETE HISTORY/BEFORE=1-Jun-1990 [Y/N] (N): Y
%DTM-S-HISTDEL, 150 history records deleted
DTM>
```

See the Command Dictionary for complete information about the DELETE HISTORY command.

---

## 3.2 DEC/Test Manager Tests

You write tests for one or more of the following reasons:

- To ensure that the introduction of a new software component does not produce a negative impact on existing software components.
- To ensure that changes to the environment outside of the application do not affect the application itself. For example, an upgraded operating system or a change in an error message file should not affect the application (other than to issue new error messages).
- To test error and boundary conditions in the application.

DEC/Test Manager supports three types of tests. Table 3–1 describes those tests.

**Table 3–1: DEC/Test Manager Test Types**

<b>Test Type</b>	<b>Description</b>
Noninteractive	A test whose template file is a DCL command procedure.
Interactive Terminal	A test that includes recorded input and output from applications in a terminal environment. You create an interactive terminal test by recording a test in a terminal environment.
Interactive DECwindows	A test that includes recorded input and output from applications in a DECwindows environment. You create a DECwindows test by recording a test in a DECwindows environment.

To create an interactive terminal test or a DECwindows test, you begin by running the software application you want to test. DEC/Test Manager records the interactive input and the application's output and places them in separate files. The file of the recorded input is called a session file, and has a default file type of .SESSION; the file of the recorded output is the benchmark file, and has a default type of .BMK.

---

## 3.3 Test Descriptions

A **test description** identifies a test and its related files to DEC/Test Manager. A test description consists of a set of fields that identify the files and other entities (filters and variables) associated with the test; it contains the information DEC/Test Manager needs to run that particular test. This section describes how to display, copy, modify, and delete test descriptions.

---

### 3.3.1 Creating Test Descriptions

You create a test description by using the `CREATE TEST_DESCRIPTION` command with qualifiers to specify the test description fields.

The least amount of information you can provide in a test description is the test name. Table 3–2 shows all the fields that you can specify in a test description. See the `CREATE TEST_DESCRIPTION` command in the Command Dictionary for more information about specifying the test description fields.

**Table 3–2: Test Description Fields**

Field	Field Type	Function
Test name	Name string	Identifies the test description.
Test prologue	File specification	Identifies a DCL command file that runs immediately before the template file. You use a prologue file to set up any special environment that the test requires. Output from a prologue file does not appear in the test results.
Test epilogue	File specification	Identifies a DCL command file that runs immediately after the template file. You use an epilogue file to clean up operations or to apply user-created filters to the result file. Unlike the prologue file, the epilogue file can directly alter the test results.

(continued on next page)



**Table 3–2 (Cont.): Test Description Fields**

<b>Field</b>	<b>Field Type</b>	<b>Function</b>
Template	File specification	Identifies a DCL command file for a noninteractive test or the session file for an interactive terminal or DECwindows test. This field defaults to test-name.SESSION for an interactive terminal or DECwindows test and test-name.COM for noninteractive tests.
Benchmark	File specification	Identifies a file that contains the expected test output. It is the standard against which DEC/Test Manager compares the results of a test run. This field defaults to test-name.BMK.
Variables	Name string and value	Identifies the variables and associated values used with the template, prologue, or epilogue files for this test.
Groups	Name string	Identifies the groups to which the test description belongs.
Test type	Boolean flags	Identifies either interactive terminal, DECwindows, or noninteractive tests.
Command	DCL command	Identifies a DCL command to be spawned when a DECwindows test is recorded or executed in a collection. This command can be used to invoke applications for inclusion in the test.
Comparison type	Value	Identifies the comparison type: screen, record, or character.
Filters	Boolean flags (One flag per filter type)	Identifies one or more filters to remove run-time dependent information from the result file.
Remark	Quoted string	Identifies a comment that you add to the history.

The following example shows you how to create an interactive terminal test and then record the session file for the test:

```
DTM> CREATE TEST_DESCRIPTION MAIL_TEST /INTERACTIVE
_Remark: Going to record a MAIL test
%DTM-I-DEFAULTED, benchmark file name defaulted to MAIL_TEST.BMK
%DTM-I-DEFAULTED, template file name defaulted to MAIL_TEST.SESSION
%DTM-S-CREATED, test description MAIL_TEST created
DTM> RECORD MAIL_TEST "Recording MAIL on the terminal"
%DTM-I-BEGIN, your interactive test session is now beginning...
Type CTRL/P twice to terminate the session.
```

§

---

### 3.3.2 Displaying Test Descriptions

Use the `SHOW TEST_DESCRIPTION` command to display a test description. The following example displays the contents of the test description `SEND_MAIL_TEST`:

```
DTM> SHOW TEST_DESCRIPTION SEND_MAIL_TEST

Test descriptions in DEC/Test Manager Library DUA0:[USER01.DTMLIB]

SEND_MAIL_TEST          "MAIL SEND command test"

  Template               = SEND_MAIL_TEST.COM
  Benchmark               = SEND_MAIL_TEST.BMK
  Prologue               = None Specified
  Epilogue               = None Specified
```

You can display more than one test by specifying test names or group names, or you can use wildcards.

Depending on the qualifiers you specify, the `SHOW TEST_DESCRIPTION` command displays the following information about tests:

- Group names
- Type of test (noninteractive, interactive, or DECwindows)
- Benchmark file specification
- Epilogue file specification
- Prologue file specification
- Template file specification
- Command string
- Filters
- Variables

You can also use qualifiers to display all the test description information or a portion of it; the default qualifier is `/INTERMEDIATE`. See the `SHOW TEST_DESCRIPTION` command in the Command Dictionary for more information about these qualifiers.

---

### 3.3.3 Copying Test Descriptions

Use the `COPY TEST_DESCRIPTION` command to create an exact or modified copy of a test description in the DEC/Test Manager library. This command enables you to create a series of similar test descriptions without repeatedly entering the `CREATE TEST_DESCRIPTION` command. You can modify some of the field values for each new test description.

The following restrictions apply to the `COPY TEST_DESCRIPTION` command:

- You cannot use wildcards.
- You cannot change the value of the benchmark field; you must use the benchmark associated with the existing test.
- When you specify the `/NOTEMPLATE` qualifier, DEC/Test Manager uses the default template file name.
- You cannot change the test type.

Although you can specify new values for some of the fields of a new test description, you must either copy a test description with its filter, variable, and group field values intact or eliminate the field values altogether; you cannot modify these values when you copy the test description. You can eliminate these field values with the `COPY TEST_DESCRIPTION` negating qualifiers. (For example, `/NOFILTER` is a negating qualifier that disassociates filters from the new test description.)

If you use a negating qualifier, DEC/Test Manager either removes the value of the qualifier or reverts the value to its DEC/Test Manager default value. See the Command Dictionary for more information about the `COPY TEST_DESCRIPTION` command, its qualifiers, and its negating qualifiers.

Example 3-1 shows how to create several similar test descriptions for a noninteractive test called `MAIL_TEST_NONINT`:

The test description generated by the `CREATE TEST_DESCRIPTION` command sets up the prologue and epilogue files. Subsequent `COPY TEST_DESCRIPTION` commands create new tests using the first test description, `MAIL_TEST_NONINT`. By default, the prologue and epilogue files that are associated with `MAIL_TEST_NONINT` are copied into the `SHOW_ALL_TEST` and `SEND_MAIL_TEST` test descriptions. However, the `MAIL_TEST_NONINT`, `SHOW_ALL_TEST`, and `SEND_MAIL_TEST` test descriptions have different template files.

If you do not specify any qualifiers, the value for the test description fields are copied from the existing test description to the new test description.

### Example 3–1: Copying Test Descriptions

---

```
DTM> CREATE TEST_DESCRIPTION MAIL TEST_NONINT -
_DTM> /TEMPLATE=MAIL_NONINT.COM/NOINTERACTIVE -
_DTM> /PROLOGUE=NOBROADCAST.COM/EPILOGUE=BROADCAST.COM
_Remark: Creating a MAIL test
%DTM-I-DEFAULTED, benchmark file name defaulted to MAIL_TEST_NONINT.BMK
%DTM-I-DEFAULTED, template file name defaulted to MAIL_TEST_NONINT.COM
%DTM-S-CREATED, test description MAIL_TEST_NONINT created
DTM>
DTM> COPY TEST_DESCRIPTION MAIL TEST_NONINT SHOW_ALL_TEST -
_DTM> /TEMPLATE=SHOW_ALL_NONINT.COM
_Remark: Copied margin test into SHOW_ALL_TEST with new template file
%DTM-I-DEFAULTED, benchmark file name defaulted to SHOW_ALL_TEST.BMK
%DTM-I-COPIED, test description MAIL_TEST_NONINT copied
-DTM-S-CREATED, test description SHOW_ALL_TEST created
DTM>
DTM> COPY TEST_DESCRIPTION MAIL TEST_NONINT SEND_MAIL_TEST -
_DTM> /TEMPLATE=MAIL_TEMPLATE.COM
_Remark: Copied margin test into SEND_MAIL_TEST with new template file
%DTM-I-DEFAULTED, benchmark file name defaulted to SEND_MAIL_TEST.BMK
%DTM-I-COPIED, test description MAIL_TEST_NONINT copied
-DTM-S-CREATED, test description SEND_MAIL_TEST created
DTM>
```

---

### 3.3.4 Modifying Test Descriptions

Use the `MODIFY TEST_DESCRIPTION` command to modify a test description to include or exclude a prologue file, an epilogue file, filters, variables, or other test description attributes.

You can add or delete all test description fields (except the template field) by using the `MODIFY TEST_DESCRIPTION` qualifiers. If you add a field that already exists, the addition overrides the current value of the existing field. For example, you can replace the current epilogue file with the epilogue file named `NODCL_BROADCAST.COM`:

```
DTM> MODIFY TEST_DESCRIPTION MAIL_TEST/EPILOGUE=NODCL_BROADCAST.COM
_Remark: Using a different epilogue file
%DTM-S-MODIFIED, test_description MAIL_TEST modified
DTM>
```

If you use a negating qualifier, DEC/Test Manager either removes the value of the qualifier or reverts the value to its DEC/Test Manager default value. See the Command Dictionary for more information about the `MODIFY TEST_DESCRIPTION` qualifiers and negating qualifiers.

Modifications remain in effect until you explicitly remove a value from the test description, or modify the test description again. The following example removes an existing prologue file specification from a test description called `MAIL_TEST`. If the prologue file exists in the DEC/Test Manager library, it is deleted; if it exists outside the DEC/Test Manager library, it is not deleted. DEC/Test Manager issues informational messages that inform you of these conditions.

```
DTM> MODIFY TEST_DESCRIPTION MAIL_TEST /NOPROLOGUE
_Remark: Deleting the prologue
%DTM-S-MODIFIED, test_description MAIL_TEST modified
DTM>
```

Only the test description fields you specify with the `MODIFY TEST_DESCRIPTION` command are affected when you modify a test description.

DEC/Test Manager ignores negating qualifiers specified for fields for which no value was assigned. For example, the `/NOPROLOGUE` qualifier is ignored if no prologue file had been previously assigned.

---

### 3.3.5 Deleting Test Descriptions

Use the `DELETE TEST_DESCRIPTION` command to delete a test description from the DEC/Test Manager library. If the test description has a benchmark file that resides in the DEC/Test Manager library, the benchmark file is also deleted. If the benchmark file is outside the library, it is unaffected.

The `DELETE TEST_DESCRIPTION` command has no effect on any result files or difference files that were produced for this test during a collection run. DEC/Test Manager deletes result and difference files with the collection rather than with the test description, because these files are associated with the collection.

The following example deletes the test description `SHOW_ALL_TEST`. Because the benchmark is in the DEC/Test Manager library, it also is deleted.

```
DTM> DELETE TEST_DESCRIPTION SHOW_ALL_TEST
_Remark: Deleting the revised Test for sending mail
Confirm deletion of test_description SHOW_ALL_TEST [Y/N] (N): Y
%DTM-S-DELETED, test_description SHOW_ALL_TEST deleted
DTM>
```

You cannot delete a test description while it is a member of any group. Use the `REMOVE TEST_DESCRIPTION` command to remove a test description from all groups to which it belongs. Then use the `DELETE TEST_DESCRIPTION` command to delete the test description.

---

## 3.4 Creating Noninteractive Tests

To create a noninteractive test in DEC/Test Manager, you must perform the following steps:

1. Write the test.
2. Write the template file.
3. Create the test description.

---

### 3.4.1 Writing a Noninteractive Test

You write a noninteractive test by invoking a text editor outside of DEC/Test Manager and creating a DCL command procedure to run the noninteractive application you want to test. The DCL command procedure in Example 3–2 issues several MAIL utility commands and sends a message to user USER01.

#### Example 3–2: Sample Noninteractive Test File

---

```
$!      *** SEND MAIL TEST ***
$!      SEND_MAIL_TEST.COM
$!
$mail
set personal "DEC/Test Manager - Project Q"
send/subject="Mail test procedure"
USER01
This test message is sent by Electronic mail using a DCL
procedure to test the MAIL utility.
$mail
show personal_name
exit
```

---

---

### 3.4.2 Writing a Template File for a Noninteractive Test

The template file for a noninteractive test can be the test itself. For example, the test in Example 3–2 can be executed by itself. If the template file is the test itself, you do not need to create a new template file to execute your test.

The template file for a noninteractive test can also be a DCL command procedure that executes the specified test and performs some action before and after the test is executed. In Example 3–3, the template file disables broadcast messages to be displayed before invoking the test and enables the broadcast messages after the test is run.

### NOTE

You can also use a test prologue file and test epilogue file to perform these actions. Chapter 6 describes using prologue and epilogue files.

### Example 3–3: Sample Noninteractive Test Template File

---

```
$!    *** MAIL TEMPLATE ***
$!    MAIL_TEMPLATE.COM
$!
$ SET BROADCAST=NONE
$ @SEND_MAIL_TEST
$ SET BROADCAST=ALL
```

---

---

### 3.4.3 Creating a Noninteractive Test Description

To create a noninteractive test description for a test, use the **CREATE TEST\_DESCRIPTION** command (**/NOINTERACTIVE** is the default qualifier), as shown in the following example:

```
DTM> CREATE TEST_DESCRIPTION SEND_MAIL_TEST/TEMPLATE=MAIL_TEMPLATE.COM
_Remark: Test for sending mail
%DTM-I-DEFAULTED, benchmark file name defaulted to SEND_MAIL_TEST.BMK
%DTM-S-CREATED, test description SEND_MAIL_TEST created
DTM>
```

The existence of files is not verified at test creation time. You can specify that the following files be associated with a test description (the template file is required):

- Template file (defaults to test-name.COM)
- Benchmark file (defaults to test-name.BMK)
- Prologue file
- Epilogue file

See the Command Dictionary for more information about the **CREATE TEST\_DESCRIPTION** command.

---

## 3.5 Creating Interactive Tests

You can record two types of tests:

- Interactive terminal tests
- DECwindows tests

### NOTE

The term **display device** indicates either a terminal screen device or a workstation screen device.

During record, DEC/Test Manager captures all input and output generated on the display device to create an interactive terminal or DECwindows test. The basic concept is the same for both terminal and DECwindows sessions; that is, DEC/Test Manager records and places the input into a template file and the output into a benchmark file.

To create an interactive terminal test description, you must specify the `CREATE TEST_DESCRIPTION` command with the `/INTERACTIVE` qualifier. To create a DECwindows test description, you must specify the `CREATE TEST_DESCRIPTION` command with the `/DECWINDOWS` qualifier.

### Session Files

A template file has a default file type of `.SESSION` for both terminal and DECwindows tests, but you can specify any file type by using the `CREATE TEST_DESCRIPTION` command with the `/TEMPLATE` qualifier. See the `CREATE TEST_DESCRIPTION` command in the Command Dictionary for more information about changing the session file type.

The stream of input that is recorded at an interactive recording session is placed in a test template file with the `.SESSION` file type. Specifically, a template file, also called a session file, contains the following data:

- A description of the type of display device on which you recorded the test
- A record of all input during the recording session
- Additional control and timing information

See Chapter 8 for more information about session files.



## Benchmark Files

A benchmark file contains the expected output for the test's execution; it is a standard by which other test results can be judged. DEC/Test Manager compares a benchmark file to the result file, which is generated using the input from a recorded session file when you run a test within a collection. A benchmark file has a default file type of .BMK.

---

### 3.5.1 Recording Tests

The RECORD command records a specified test. If you do not specify an input file, DEC/Test Manager records input from the display device, keyboard, and pointer device.

If you specify an input file on the RECORD command line, DEC/Test Manager records from the input file. Chapter 8 describes terminal input files. Chapter 9 describes DECwindows input files.

---

#### 3.5.1.1 Recording Key Sequences

DEC/Test Manager provides two sets of recording key sequences, depending on the type of test you are recording. During a recording session, these key sequences can be used to mark screens for comparison, or to terminate the recording session.

To enter record key sequences for terminal tests, first type the termination character (the default termination character is ^P (CTRL/P)), then type a valid command character. If you type an invalid command character, the terminal bell sounds once (unless it is disabled). To override the default termination character, use the /TERMINATION\_CHARACTER qualifier on the RECORD command.

To enter record key sequences for DECwindows tests, press the command keySYM (key symbol) key in conjunction with a valid command character. A **keySYM** is a value associated with a key on the keyboard. The default command keySYM key is the Compose key. The workstation bell sounds once when the execution of a valid command completes. If you type an invalid command character, the workstation bell sounds three times (unless it is disabled).

To override the default command keySYM key, use the /KEYSYM qualifier on the RECORD command. Table 3-3 shows the recording key sequences for both terminal and DECwindows tests.

**Table 3–3: Recording Key Sequences**

<b>Terminal Test</b>	<b>DECwindows Test</b>	<b>Function</b>
CTRL/P-B		Starts automatic screen comparison and ends manual screen comparison for terminal-based tests.
CTRL/P-!		Invokes an editor so that you can enter a comment into the session file. You end the comment by pressing CTRL/! in a terminal session file.
CTRL/P-E		Ends automatic screen comparison and begins manual screen comparison for terminal-based tests.
CTRL/P-CTRL/P	Compose/P	Ends the recording session and returns control to the previous command level. When you end a recording session this way, DEC/Test Manager saves the session and benchmark files.
CTRL/P-?		Displays the current screen comparison mode and lists the available recording functions.
CTRL/P-I		Inserts an input file into the session file you are recording. DEC/Test Manager prompts you for the input file specification.
CTRL/P-M	Compose/M	Marks a screen for comparison. DEC/Test Manager automatically compares all screens for terminal-based tests. This control key sequence is used in terminal-based tests when automatic comparison is disabled. This is the only way to mark screens for DECwindow tests.
CTRL/P-CTRL/Z	Compose/Z	Ends an interactive recording session and saves the session file but does not save the benchmark file.

(continued on next page)

**Table 3–3 (Cont.): Recording Key Sequences**

<b>Terminal Test</b>	<b>DECwindows Test</b>	<b>Function</b>
CTRL/P-W		Invokes a prompt for you to specify a wait time. You issue this control key sequence at the place in a session file that you want the wait to occur. When the session file is subsequently played or executed, the wait time is included in the session file.
CTRL/P-CTRL/C	Compose/C	Aborts an interactive recording session without saving any of the generated files.
	Compose/S	Starts a DECwindows recording session.

### 3.5.1.2 Exiting from a Recording Session

To exit from a terminal recording session, press CTRL/P twice or press Compose/P for a DECwindows recording session. Control returns to the DCL level if you recorded a test from the DCL prompt, to the DEC/Test Manager level if you recorded a test from the DTM> prompt, or to the DEC/Test Manager DECwindows interface.

### 3.5.1.3 Redefining the Termination Character

The default termination character for interactive terminal tests is CTRL/P; the default for DECwindows tests is Compose/P. To redefine the termination character, you use the RECORD command with the /TERMINATION\_CHARACTER qualifier.

The termination character can be any single character. To specify a control key sequence, enter a circumflex (^) followed by the character you want to use. For example, to enter the termination character ^D (CTRL/D), enter a circumflex followed by a D, as shown in the following example:

```
DTM> RECORD MAIL_TEST/TERMINATION_CHARACTER=^D ""
Type CTRL/D twice to terminate the session.
.
.
.
```

You can also specify a termination character with a decimal value that translates into an ASCII character. For example, decimal 12 translates to ^L (CTRL/L); using 12 produces the following results:

```
DTM> RECORD MAIL TEST/TERMINATION_CHARACTER=12 ""
Type CTRL/L twice to terminate the session.
.
.
.
```

---

### 3.5.1.4 Redefining the Command Keysym Key

You can redefine the command keysym key for a DECwindows test by using the RECORD command with the /KEYSYM qualifier.

The command keysym key must be in the DECwindows Latin-1 keysym encodings. Display the file DECW\$INCLUDE:KEYSYMDEF.H with the DCL TYPE command to view a listing of the Latin-1 keysym keys.

You can specify a keysym key on the /KEYSYM qualifier by entering its decimal or hexadecimal encoding. For example, to use CTRL as the command key symbol, enter the following command:

```
DTM> RECORD test-name/KEYSYM=0xffe3
```

The following sections describe recording options that you can use in various combinations to tailor your recording environment. See Chapter 6 for more information about tailoring the test system.

---

## 3.5.2 Interactive Terminal Recording

The template file for an interactive terminal test is the recorded terminal session file that DEC/Test Manager produces when you use the RECORD command. An interactive terminal test requires user input from a terminal keyboard. For example, an interactive terminal test could be a forms program that requests information from you.

DEC/Test Manager subsequently uses the recorded session file to supply data to applications in future test runs to ensure that the applications you are testing have not regressed. You can have DEC/Test Manager execute the interactive session file on the screen or in batch mode. Either way, DEC/Test Manager supplies the input data that was recorded.

If your interactive terminal session will need a screen size larger than 24 lines by 132 columns, set the display device to the largest size it will need during recording before you begin the recording session.

Example 1–1 in Chapter 1 shows the recording of an interactive terminal session.

---

### 3.5.3 Interactive DECwindows Recording

An interactive DECwindows test records DECwindows environment input from the keyboard and from the pointer device. Tests performed in the DECwindows environment are recorded by specifying the DEC/Test Manager RECORD command or by pulling down the Testing menu, choosing the Record menu item, and choosing the DECwindows... submenu item. DEC/Test Manager uses the recorded session file to supply the input data when you subsequently test an application or applications, just as with interactive terminal tests.

Before recording a DECwindows test, you should ensure that the conditions of the test at its start are reproduced when the test is executed. This enables you to run a DECwindows session file repeatedly in one play back, without dependencies on an initialized workstation.

For example, iconizing all windows and ordering the icons at the start of a test enables you to reproduce the start condition of a test. It also enables you to mask out small portions of the workstation screen using the mask editor. See Section 2.2.8 for information about masking portions of the workstation screen.

To significantly reduce the size of benchmark and result files, set the workstation background to a solid color during recording and testing. DEC/Test Manager uses compression techniques to minimize the size of a benchmark file.

When marking screens for comparison on a color workstation, DEC/Test Manager converts the screen image to bitonal. To ensure proper image comparison, the colors displayed on the screen must be of sufficiently high and low intensity (bright and dark) such that they are converted to black and white as expected.

You can invoke an application or run a command file at the start of a DECwindows recording session by entering a command in the Command field in the Create DECwindows Test dialog box; you can also use the CREATE TEST\_DESCRIPTION command with the /DECWINDOWS and /COMMAND qualifier.

## NOTE

There is no automatic comparison in a DECwindows recording session. Screens must be explicitly marked for comparison.

See Chapter 2 for a sample interactive DECwindows recording session.

---

### 3.6 Creating an Input File from a Session File

An **input file** is a textual representation of a session file that you can edit by using the text editor of your choice. The **EXTRACT** command extracts an input file from an interactive terminal or DECwindows session file without altering the session file.

You can create a new session file, using an input file, by specifying the **/INPUT** qualifier with the **RECORD** command. The **/INPUT** qualifier specifies the input file to be read.

See Chapter 8 and the Command Dictionary for complete information about using the **EXTRACT** command to create input files from terminal session files.

See Chapter 9 and the Command Dictionary for complete information about using the **EXTRACT** command to create input files from DECwindows session files.

---

### 3.7 Playing Back an Interactive Test

DEC/Test Manager provides two methods for playing back an interactive terminal or DECwindows test:

- Use the **PLAY** command to playback a session file interactively.
- Use the **CREATE COLLECTION** and **RUN** commands to execute a test.

The **PLAY** command executes a specific session file. The file is not played back as part of a collection and the results of the playback are not compared. The **RUN** command executes a collection of tests interactively; results may be compared.

Chapter 4 describes how to create collections and run tests.

---

### 3.7.1 Playing an Interactive Terminal Session

To play an interactive terminal session file, use the **PLAY** command with the **/INTERACTIVE** qualifier. For example:

```
DTM> PLAY MAIL_TEST.SESSION/INTERACTIVE
%DTM-I-BEGIN, your interactive test session is now beginning
.
.
.
%DTM-S-CONCLUDED, your interactive test session has concluded
DTM>
```

A session file is executed as if it was being run on the same type of display device on which it was recorded. If the display device characteristics differ from those for the recording display device, the output may not appear as you would expect.

---

### 3.7.2 Playing an Interactive DECwindows Session

To play an interactive DECwindows session file, use the **PLAY** command with the **/DECWINDOWS** qualifier. For example:

```
DTM> PLAY MAIL_TEST.SESSION/DECWINDOWS
.
.
.
DTM>
```

If the test corresponding to the session file has a **DCL** command associated with it, specify the **DCL** command with the **/COMMAND** qualifier.

#### **NOTE**

To play a DECwindows session file, DEC/Test Manager requires a physical workstation, and its DECwindows server. Do not use the mouse or keyboard while a DECwindows session file is being played. This may cause the session file to be played incorrectly.

---

## 3.8 Processing Considerations for Interactive Terminal Tests

The following sections describe processing considerations and restrictions on the types of interactive terminal applications you can test when using DEC/Test Manager.

---

### 3.8.1 Time-Dependent Applications

DEC/Test Manager cannot execute the test at the same speed at which it was recorded; therefore, time-dependent screens marked for comparison usually do not match, and the comparison of the test results usually are unsuccessful. For example, you cannot test the VMS Phone Utility because your input as the person initiating the call depends on the person you are calling answering your call. These two time-dependent events cannot be consistently duplicated.

Other examples of timing-dependent applications are those that are submitted as a batch job or executed as a subprocess. You cannot test timing-dependent applications unless the application is a submitted batch job with the DCL SYNCHRONIZE command. Using the SYNCHRONIZE command, you have the choice of either waiting for the batch job or subprocess to finish or performing other operations with the application. The VAX Language-Sensitive Editor (LSE) COMPILE command is an example of this type of timing-dependent application.

---

### 3.8.2 CTRL/C or CTRL/Y

While recording a test, you should not press CTRL/C or CTRL/Y except at a point where the application being tested is expecting input. Pressing CTRL/C or CTRL/Y at a point other than where the application is expecting input terminates the application at a random point that generally cannot be duplicated when you run the test. For example, if you press CTRL/C or CTRL/Y while output is being displayed during testing of the DCL DIRECTORY command, you create a screen that cannot be consistently duplicated; the comparison of the test results is usually unsuccessful.

---

### 3.8.3 Type-Ahead

If you must test an application built on a tool that uses the type-ahead feature (such as the VAX Text Processing Utility (VAXTPU) or the VAX Language-Sensitive Editor (LSE)), set your terminal to /NOTYPEAHEAD before you record the interactive terminal session. DEC/Test Manager does not support the testing of interactive applications that behave differently when you type ahead. Recorded terminal sessions are always played back without the type-ahead feature. Thus, if you record a test with the type-ahead feature, DEC/Test Manager plays it back without that feature and the comparison of the test results usually is unsuccessful.



You do not need to set your terminal to `/NOTYPEAHEAD` when you play back the recorded terminal session or when you read from an input file.

---

### **3.8.4 Applications That Accept Unsolicited Input**

You can use DEC/Test Manager to test only those applications that accept input after prompting for it. You cannot test programs such as the VMS Monitor Utility (MONITOR) with DEC/Test Manager. MONITOR displays screen after screen of continuously changing statistical information about the system. After you invoke MONITOR, it does not prompt you for input; it displays information until you terminate it by pressing CTRL/Z. The termination occurs in a way that cannot be consistently duplicated; thus, the comparison of the test results usually is unsuccessful.

---

### **3.8.5 Device Type and Terminal Characteristics**

If DEC/Test Manager does not recognize the device type of the terminal you are using to record the interactive terminal session, it defaults to a VT100-compatible terminal. Chapter 8 describes the significance of device type and terminal characteristics when recording an interactive terminal session.

---

## **3.9 Processing Considerations for DECwindows Tests**

The following sections describe processing considerations and restrictions on the types of DECwindows applications you can test when using DEC/Test Manager.

---

### **3.9.1 Playing DECwindows Tests**

Tests played in a DECwindows environment play at a different rate than those played in an interactive terminal environment.

During interactive terminal test playback, session files are generally played at a rate faster than the rate at which they were recorded. This is because DEC/Test Manager detects when an application is ready for more input, and sends session file input to the application as fast as it will accept it.

However, during DECwindows testing, DEC/Test Manager does not detect when an application is ready for more input. Thus, DECwindows sessions are played in real time. When a session file is played, keyboard and mouse movements are played back at the same speed at which they were recorded. For example, if you press a key at 10-second intervals while recording a test, the keypress events are repeated at 10-second intervals when you play the session file.

DECwindows sessions are played in real time whether the session is played as part of a test execution (using the RUN or SUBMIT commands) or played separately (using the PLAY command).

### **CAUTION**

Because a DECwindows test is played in real time, it may be affected by other processes that may degrade system performance.

You must ensure that no unanticipated loads are placed on the system while a DECwindows session file is playing, or you may not get the desired results. For example, to test the program Calendar, use the following steps:

1. Select the Calendar entry from the Session Manager Applications menu.
2. When the Calendar appears on the screen (after approximately 10 seconds) use its features as desired as part of the test recording.
3. Complete the test recording, initialize your environment for the test, and start playing the session file.

Suppose you have another active process that is compiling at the same time as the test play. When you play the session file, the Calendar may take 15 seconds to appear instead of the 10 seconds it took to appear when the test was originally recorded. Because the DECwindows session file is playing in real time, the keyboard and mouse input that occur in the Calendar program during the test recording session is sent to the workstation before the Calendar appears on the screen and is able to accept input. The input is then lost and the session file fails to play as expected.

You can work around real time restrictions by applying these solutions:

- Use a dedicated system for DECwindows testing. Do not run other programs that use CPU cycles during the DECwindows test.
- Extract an input file from a DECwindows session file (using the EXTRACT command) and enter synchronization points into the DECwindows Session, or change the timing factors in keyboard or mouse movement records so that the test will play at a slower rate in critical sections (see Chapter 9 for more information on extracting from session files).

---

### **3.9.2 Storing DECwindows Benchmark and Result Files**

Screen image data stored in DECwindows benchmark and result files requires significant disk storage space. DEC/Test Manager compresses the image data to save storage space. However, DEC/Test Manager cannot successfully compress screen images where a patterned background is used (for example, the DECwindows default background). You should customize your screen background to display a solid color before invoking DEC/Test Manager in the DECwindows environment.

---

### **3.9.3 Environment Initialization**

You must ensure that the DECwindows environment is in the same state for a play as it was when the session file was recorded. Factors which may affect the environment include applications with windows displayed and window positions.



# Organizing and Executing Test Collections

---

This chapter describes how to use DEC/Test Manager to organize and run test collections; it provides information on the following topics:

- Selecting tests to execute and organizing them into collections
- Executing collections in batch mode
- Executing collections interactively at a terminal or DECwindows workstation
- Stopping collections
- Displaying the collection summary
- Deleting collections
- Re-creating collections
- Comparing test results

---

## 4.1 Creating Collections

A **collection** is a snapshot of specified test descriptions and the DEC/Test Manager library as they exist at the time you create the collection. You must organize tests into collections before you can execute them to produce result files for comparison.

Collections are stored in a DEC/Test Manager library. They can have prologue files, epilogue files, and variables associated with them, though the prologue files and epilogue files must be stored in locations other than the DEC/Test Manager library (VAX DEC/Code Management System (CMS) libraries, for example; see Chapter 7).

As test descriptions change, you must re-create the collection to reflect those changes. You can include a test description in more than one collection.

You create a collection by using the `CREATE COLLECTION` command, which generates a command file that accesses the specified set of tests and their related files.

When you create a collection, DEC/Test Manager reads each test description you specified; it then runs each test in the collection alphabetically by test name.

Collection names cannot begin with the characters `DTM$`. The test group expression selects the tests to be placed in the collection. Valid test group expressions are test names, group names (to indicate that all the test descriptions in the group should be part of the collection), or a list containing any combination of these. You must use the `/GROUP` qualifier to label group names and group expressions in a test group expression. You can use wildcards.

The following example shows how to create the collection `MAIL_COLL` and include all tests that begin with the string `MAIL`:

```
DTM> CREATE COLLECTION MAIL_COLL MAIL* "Tests of MAIL commands"  
%DTM-S-CREATED, collection MAIL_COLL created
```

If you modify test descriptions or the contents of groups after including them in a collection, those changes are not reflected in the collection unless you re-create it using the `RECREATE` command. For example, if you delete the `MAIL_SHOW_ALL_TEST` test description after you create the `MAIL_COLL` collection, the `MAIL_COLL` collection still contains pointers to the files associated with `MAIL_SHOW_ALL_TEST`. If you then try to run this collection, it will not run properly because DEC/Test Manager will try to find the deleted test.

You do not have to re-create a collection for benchmark files because they may be updated; this is the only exception. Section 4.5 describes the `RECREATE` command.

When you create a collection, DEC/Test Manager attempts to resolve all the file specifications in each test description and any file specifications included with the `CREATE COLLECTION` command. If DEC/Test Manager fails to find one or more of the required files, it lists the files and does not create the collection. However, if the only file that DEC/Test Manager cannot find is the benchmark file for a test, DEC/Test Manager creates the collection and treats the test with the missing benchmark file as a new test.

If you specify the `CREATE COLLECTION` command with the `/NOVERIFY` qualifier, DEC/Test Manager creates the collection but resolves only those file specifications included on the command line.

Collection prologue and epilogue files are command files that are run before and after (respectively) a collection is executed. They may be used to set up an environment for the entire collection; they are similar to test prologue and epilogue files. DEC/Test Manager also provides collection-wide variables to tailor the test environment (see Chapter 6).

The following example creates a collection containing one test description and immediately submits it to the batch queue:

```
DTM> CREATE COLLECTION RUN_MAIL RMTEST/NOVERIFY/SUBMIT=(NOTIFY) -  
_DTM> /PROLOGUE=SETUP.COM "First run of test RMTEST"
```

- The first parameter identifies the collection name as RUN\_MAIL.
- The second parameter, RMTEST, identifies the test you want to include in the collection and to run.
- The /NOVERIFY qualifier specifies that DEC/Test Manager is to create the collection without verifying the existence of any files associated with the test description.
- The /SUBMIT qualifier specifies that DEC/Test Manager is to submit the collection to the batch queue as soon as the collection is created. The NOTIFY keyword specifies that you will be notified when the batch job has completed.
- The /PROLOGUE=SETUP.COM qualifier specifies that SETUP.COM is the collection prologue. Because the collection prologue file SETUP.COM is issued with the CREATE COLLECTION command (it is associated with the collection), DEC/Test Manager verifies the existence of this file even if you specify the /NOVERIFY qualifier.

---

## 4.2 Executing Collections

When you execute a collection, DEC/Test Manager sets up the test environment and executes all the tests in the collection. Each test in a collection generates a separate result file. The result file contains the output generated by the template file. The result file is used for comparison against a test's benchmark file. Section 4.6 describes comparing test results. See Chapter 2 for information on executing collections in a DECwindows environment.

DEC/Test Manager associates the result file with a test by adding its name to the **result description** for the test. Result descriptions are described in Chapter 5.

You can have DEC/Test Manager execute collections interactively or in batch. Although DECwindows tests may be executed in batch mode, they still require a physical workstation and its DECwindows server in order to be played. See Chapter 9 for more information on playing DECwindows tests.

### NOTE

When you interactively execute the tests in a large collection, you can tie up the display device for a long time. Consider executing large collections in batch; the results are the same, provided the batch and interactive environments are the same.

If your login command file sets up your interactive environment differently than it sets up your batch environment, the results may vary for the same test executed both interactively and in batch mode. For more information, see the *Guide to Using DCL and Command Procedures on VMS*.

All tests in a collection are executed during the collection run. After the collection has executed and has been compared, you can examine the test results by using the Review subsystem (see Chapter 5).

When you submit a collection for execution in batch, or execute it interactively, DEC/Test Manager performs the following tasks:

1. Defines the `DTM$COLLECTION_NAME` variable (defined by DEC/Test Manager as the collection name)
2. Defines any global variables
3. Executes the collection prologue file, if one exists
4. Calls a generic command file in the DEC/Test Manager library that performs the following tasks on each test:
  - a. Defines any local variables
  - b. Defines the `DTM$TEST_NAME` logical name (defined by DEC/Test Manager as the test name of the current test)
  - c. Executes the test prologue, if one exists
  - d. Executes the test template
  - e. Defines the `DTM$RESULT` logical name (defined by DEC/Test Manager as the test result file name of the current test)
  - f. Executes the test epilogue, if one exists
  - g. Executes DEC/Test Manager-provided filters, if any are associated with the test
  - h. undefines local variables



- i. Undefined the DTM\$RESULT logical name
5. Compares the result file with the benchmark file
6. Executes the collection epilogue, if one exists

---

## 4.2.1 Executing Collections in Batch

DEC/Test Manager provides two ways to execute a collection in batch:

- Use the CREATE COLLECTION command with the /SUBMIT qualifier to automatically submit the collection to the batch queue after creating it. You can also specify any of the SUBMIT command qualifiers.
- Use the SUBMIT command to submit the collection.

If you use the CREATE COLLECTION command with the default qualifier /NOSUBMIT, you must use the SUBMIT command to execute the collection in batch. The following example shows how to submit a collection:

```
DTM> SUBMIT MAIL_COLL/NOTIFY/LOG_FILE=[]/QUEUE=SYS$LARGE
%DTM-S-SUBMITTED, collection MAIL_COLL submitted
-DTM-I-TEXT, Job MAIL_COLL (queue SYS$LARGE entry 1000) started on
SYS$LARGE
```

See the Command Dictionary for more information about the SUBMIT command and its qualifiers.

You can submit a collection to a batch queue more than once. However, if you attempt to resubmit a collection that you have executed and not reviewed, DEC/Test Manager prompts you to confirm that you want to resubmit the collection without reviewing it. To eliminate the confirmation prompt when resubmitting a collection without reviewing it, specify the SUBMIT command with the /NOCONFIRM qualifier.

---

## 4.2.2 Executing Collections Interactively

You can execute a collection interactively by using the RUN command. The collection you execute can contain any combination of noninteractive, interactive terminal and DECwindows tests.

If you specify a test name rather than an existing collection name as the parameter for the RUN command, DEC/Test Manager prompts you for automatic collection creation. Collections created this way contain only the specified test and have the same collection name as the test name.

The RUN command displays the output of each test on the screen. If you want to see messages from the prologue or epilogue file, specify the /LOG\_FILE qualifier.

The following example runs the collection SEND\_MAIL\_COLL on your display device. (Example 3-2 in Chapter 3 shows the SEND\_MAIL\_TEST test.) The output of this test is as follows:

```
DTM> RUN SEND_MAIL_COLL "running the send mail test"
Starting SEND_MAIL_TEST test run...

Your personal name is "DEC/Test Manager - Project Q"

Performing post-run cleanup with comparison...

%DTM-I-NEWTEST, test SEND_MAIL_TEST is a New test
%DTM-S-COMPARED, collection SEND_MAIL_COLL compared
DTM>
```

Section 4.6 describes how to compare the results of the execution of tests in a collection with the benchmark files of each test in a collection. See Chapter 9 for information about playing DECwindows tests.

---

### 4.2.3 Stopping the Execution of Collections

Use the STOP command to terminate a collection executing in batch; this command stops execution of the collection and cleans up the DEC/Test Manager library.

Press CTRL/C (rather than CTRL/Y) to terminate a collection running interactively.

If you stop an executing collection with a command other than the STOP command or a CTRL/C, or if the system crashes while a collection is executing, errors will occur and you will not be able to review the collection. Pressing CTRL/C or typing the STOP command to terminate a collection run allows DEC/Test Manager to restore its library to a consistent state and to perform necessary post-run cleanup after the collection run stops. See Chapter 7 for instructions on how to recover the library, review the executed tests, and rerun the tests that did not execute.

The following example stops the execution of the collection MSGTEST. Note that the /CONFIRM qualifier is specified to have DEC/Test Manager issue a confirmation message (/NOCONFIRM is the default qualifier).

```
DTM> STOP MSGTEST /CONFIRM "Stopping collection run"
Confirm stop of collection MSGTEST [Y/N] (N): Y
%DTM-S-STOPPED, collection MSGTEST stopped
DTM>
```

---

## 4.3 Displaying a Collection Summary

Use the `SHOW COLLECTION` command to display a brief listing of the attributes of a collection.

You can specify qualifiers to determine the amount of information to be displayed about a collection. With the default `/INTERMEDIATE` qualifier, the collection summary displays the following information:

- The collection name
- The number of tests in the collection
- The time the collection was created
- The command that created the collection and the remark associated with it
- The collection's status—whether it has been run, compared, reviewed, rerun, or stopped
- The status of the tests in the collection—how many are successful, unsuccessful, new, updated, not run, or whose comparison aborted

The following example displays a summary of information for the collection `MSGTEST`.

```
DTM> SHOW COLLECTION MSGTEST
Collections in DEC/Test Manager Library DUAO:[USER01.DTMLIB]

MSGTEST      4 test      31-MAY-1988      11:09:17 "Group of message tests"
              Command: CREATE COLLECTION MSGTEST INFOMSGTEST/GROUP
                                "Messages"
              Status: has been run, compared, not reviewed
              Successful count: 0      Unsuccessful count: 0
              New test count: 4       Updated test count: 0
              Test not run count: 0    Comparisons Aborted: 0

DTM>
```

See the Command Dictionary for more information about the `SHOW COLLECTION` command and its qualifiers.

---

## 4.4 Deleting Collections

Use the `DELETE COLLECTION` command to delete a collection. You should delete a collection after you review it and no longer need the results. When you delete a collection, all collection-related files are deleted. Benchmark files, test descriptions, and groups, including any groups created during a Review session, are not deleted.

You can delete several collections at a time by using wildcards, or use commas to separate collection names in a list.

You cannot delete a collection while it is being run or while it is in use.

When you issue the **DELETE COLLECTION** command, DEC/Test Manager displays each collection name before it is deleted and requests you for confirmation. The following example deletes the collection **MSGTEST**.

```
DTM> DELETE COLLECTION MSGTEST "no longer needed"
Confirm deletion of collection MSGTEST [Y/N] (N): Y
%DTM-S-DELETED, collection MSGTEST deleted
DTM>
```

#### **NOTE**

Your ability to delete a collection depends on the protection of the files in the collection and your VMS privileges. See Section 7.3 for more information about file protection.

---

## **4.5 Re-creating Collections**

Use the **RECREATE** command to re-create a collection. You will need to re-create a collection if you have changed one or more of the files required by tests in the collection, or if you have changed other information in the library since you created the collection. You cannot re-create a collection after you have deleted it.

#### **CAUTION**

Ensure that the prologue and epilogue files associated with the collection and template, prologue, and epilogue files associated with the tests in the collection exist in their proper directories. If DEC/Test Manager cannot find the template, prologue, or epilogue files specified on the original **CREATE TEST\_DESCRIPTION** command, the original collection is deleted but a new collection is not created.

When you create a collection, DEC/Test Manager stores the command in the DEC/Test Manager database. When you re-create a collection using the **RECREATE** command, DEC/Test Manager uses the command stored in the database so that none of the file specifications or qualifiers are changed.

---

## 4.6 Comparing Test Results

For every test in a collection that runs to completion, DEC/Test Manager compares the result file against the benchmark file (if it exists), saves the comparison status for the test, and saves any differences in a difference file. If the benchmark file and result file match, DEC/Test Manager deletes the result file.

If you specify the `CREATE COLLECTION` command with the `/NOCOMPARE` qualifier, or if a collection is only partially run, you must use the `COMPARE` command to manually compare the existing result files with the appropriate benchmark files.

A collection may only partially run for one of the following reasons:

- You press `CTRL/C` to abort an interactive execution
- You use the `STOP` command to stop a collection you submitted to batch
- DEC/Test Manager terminates abnormally

You must compare the results of a test before you can review them. You can specify one of three types of test comparisons:

- Screen comparison (using the `COMPARE` command with the `/SCREENS` qualifier; available for interactive terminal and DECwindows tests only)
- Record comparison (using the `COMPARE` command with the `/RECORDS` qualifier; available for noninteractive and interactive terminal tests only)
- Character comparison (using the `COMPARE` command with the `/CHARACTERS` qualifier; available for noninteractive and interactive terminal tests only)

Table 4–1 lists the comparison status reported by DEC/Test Manager for each test.

**Table 4–1: Comparison Status Values**

<b>Comparison Status</b>	<b>Meaning</b>
Comparison aborted	A test whose comparison could not be completed. The benchmark file exists, but the result file and difference file might not.

(continued on next page)

**Table 4–1 (Cont.): Comparison Status Values**

<b>Comparison Status</b>	<b>Meaning</b>
New test	A test that does not have a benchmark file. A result file was produced, but no difference file exists.
Not run	A test that did not run in a partially run collection. A benchmark file might exist. No result file or difference file exists.
Successful	A test whose benchmark and result files match. The result file has been deleted and no difference file exists.
Unsuccessful	A test whose benchmark and result files do not match. A difference file exists.
Updated	A test whose benchmark file has been updated after the comparison has been performed for a collection. No result file or difference file exists.

You can use the `/IGNORE` qualifier to specify types of special characters that DEC/Test Manager should ignore during a comparison. You can use the `/FULL` qualifier to specify that both identical and different data be included in the comparison report of the difference file.

See the Command Dictionary for complete information about the `COMPARE` command and its qualifiers.

After running and comparing a collection, you can review the test results and the comparison statuses with DEC/Test Manager. Reviewing a collection gives you access to the results of a collection run and to other collection information. The Review subsystem is described in Chapter 5.

---

## 4.7 Recomparing Partially Compared Collections

If the comparison of a collection is terminated abnormally before comparing the whole collection, you can recompare the collection using the `COMPARE` command.

DEC/Test Manager does not recompare previously compared tests that had a successful comparison status.

### NOTE

If you need to recompare a collection, do not review the partially compared collection. You cannot compare a collection that has been reviewed.

# Reviewing Test Results

---

This chapter shows you how to review test results in the terminal environment and in the DECwindows environment; it provides information on the following topics:

- Review concepts
  - Output files
  - Comparison statuses
  - Specifying result descriptions
- Examining test results
  - Using the Review subsystem
  - Displaying test results
  - Printing test results
- Working with test results
  - Updating and Creating benchmark files
  - Reviewing partially run collections

---

## 5.1 Review Concepts

When you execute a collection, DEC/Test Manager compares the test results with the benchmark file for each test that has been run. If the comparison is unsuccessful (differences are detected), DEC/Test Manager creates a difference file. If the comparison is successful (no differences are detected), then the result file is deleted and no difference file is created.

You can then review the differences using the Review subsystem, which gives you access to the results obtained by executing the collection, as well as to other information about the collection. It also enables you to invoke the VAX Performance and Coverage Analyzer (PCA) to gather performance and coverage data for the test. Section 5.2 describes the Review subsystem. *Using VAXset* describes using DEC/Test Manager with PCA.

You must execute and compare a collection before it can be reviewed. See Section 5.3.3 for instructions for reviewing a partially run collection.

In a DECwindows environment, you can review test results by direct manipulation in the collection view. However, if you need to update benchmark files, you must pull down the Testing menu, choose the Review menu item, and choose the Open menu item to lock the collection so that only you can perform an update operation.

This section describes result descriptions, test output files, and comparison statuses: concepts that apply to both the terminal and the DECwindows environments.

---

## 5.1.1 Using Result Descriptions

Result descriptions contain information about test output and comparison statuses. In the same way that a test description contains information about test files, DEC/Test Manager creates a result description that summarizes information about the test results (see Section 5.1.1.1).

The **result description name** for a test is the same as its test name. Each result description corresponds to a test description. A result description contains the following information:

- The result description name.
- The comparison status of the test.
- Whether the output files exist; if the benchmark file exists, DEC/Test Manager displays its file specification.

The following example shows the result description format in a terminal environment.

```
DTM_REVIEW> SHOW MAIL_TEST
Result Description MAIL_TEST          Comparison Status : Successful

    Benchmark File MAIL_TEST.BMK
    Result file does not exist
    Difference file does not exist

DTM_REVIEW>
```



---

### 5.1.1.1 Output Files

After DEC/Test Manager records, or executes and compares a collection, it may generate one to three output files, depending on its comparison status. Table 5–1 shows the possible output files.

**Table 5–1: DEC/Test Manager Output Files**

---

<b>Output File</b>	<b>Description</b>
Benchmark	Contains the expected output for the test.  For an interactive terminal test that has a benchmark file, DEC/Test Manager also creates the file test-name.BMK_SCREEN. This file contains printable copies of the interactive screen images corresponding to the information in the benchmark file.
Result	Contains the results of a test's execution within a collection.  For an interactive terminal test that has a result file, DEC/Test Manager also creates the file test-name.RES_SCREEN. This file contains printable copies of the interactive screen images corresponding to the information in the result file.
Difference	Contains the differences between the benchmark and result files.  This file is created during comparison of the benchmark and result files. A difference file is created only if differences exist between the benchmark and result files.

---

---

### 5.1.1.2 Comparison Status

For every test in a collection that runs to completion, DEC/Test Manager compares the result file with the benchmark file (if it exists) and reports the result as the comparison status for the test. Chapter 4 describes comparison statuses. They are listed briefly as follows:

- Comparison aborted
- New test
- Not run
- Successful
- Unsuccessful
- Updated

---

## 5.1.2 Specifying Result Descriptions

When reviewing the results for a collection of tests in the Review subsystem you should note the following general concepts:

- You can specify each result description both by its result description name and by its comparison status. You can organize sets of result descriptions that have similar characteristics in the following ways:
  - By specifying a result description expression (result description names containing wildcards). For example:  

```
DTM_REVIEW> SHOW MAIL*
```
  - By specifying one or more comparison status qualifiers. For example:  

```
DTM_REVIEW> SHOW/UNSUCCESSFUL
```
  - By specifying a result description expression with one or more comparison status qualifiers. For example:  

```
DTM_REVIEW> SHOW MAIL* /UNSUCCESSFUL
```
- For the SHOW and PRINT commands, you can also specify output file qualifiers to print or display output files. For example, you can enter the SHOW \*/SUCCESS/BENCHMARK command to display the benchmark files for all successful tests.
- When you enter a Review subsystem command that accepts the result description expression and the comparison status qualifiers, the information in Table 5–2 applies.

**Table 5–2: Result Descriptions and Comparison Status Qualifier Variations for the SHOW Command**

<b>Result Descriptions</b>	<b>Comparison Status Qualifiers</b>	<b>Result of the Command</b>
None	None	Shows the current result description.
None	One or more	Shows all result descriptions with the specified comparison statuses; DEC/Test Manager interprets the result description parameter of the SHOW command as the wild-card parameter. For example, you can enter SHOW/FILES /SUCCESS/NEW to display the comparison status and to display whether output files exist for all successful and new result descriptions. The current result description position remains unchanged when you specify the SHOW command without specifying a result description.
One	None	Shows the specified result description and makes it the current result description. For example, you can enter SHOW RMTEST to display the result file for result description RMTEST and then make RMTEST the current result description.
One	One or more	Shows the specified result description, makes it the current result description, and ignores the comparison status qualifiers. For example, you can enter SHOW/SUCCESS RMTEST to display the result file for result description RMTEST and then make RMTEST the current result description. DEC/Test Manager ignores the /SUCCESS qualifier with this command and displays the result file.
One or more (using wild-cards)	None	Shows all result descriptions matching the result description expression, but it does not change the review position of the current result description. For example, you can make the current position the RMTEST result description and then enter SHOW /RESULT *RM* to display the result files for all result descriptions matching *RM*; the review position remains on RMTEST.
One or more	One or more	Shows all result descriptions that match both the result description expression and any of the comparison status qualifiers. For example, if you enter INSERT/UNSUCCESS /NOT_RUN *RM*, all test descriptions that match *RM* and have the unsuccessful or not run comparison status are marked for insertion into a group when you exit from the Review subsystem.

---

## 5.2 Examining Test Results

The procedure for examining test results after executing test collections is summarized in the following steps:

1. Invoke the Review subsystem.
2. Examine the collection summary information.
3. Examine the results for each test.
4. Examine one or more of the output files referenced in the result description: the result, difference, or benchmark files.

You can examine test results by issuing the `SHOW` and `PRINT` commands. The `SHOW` command displays information about result descriptions and displays their output files on the display device. The `PRINT` command prints copies of specified output files.

---

### 5.2.1 Using the Review Subsystem

This section provides an overview of the Review subsystem and shows you how to use it to locate test results in a terminal environment. The Review subsystem exists only in a terminal environment; DECwindows tests cannot be reviewed in a terminal environment. You review DECwindows test results using views in the DECwindows environment (see Chapter 2).

---

#### 5.2.1.1 Review Subsystem Overview

The `REVIEW` command invokes the Review subsystem, which you use to examine the test results of a collection execution. In a terminal environment, the Review subsystem is indicated by the `DTM_REVIEW>` prompt.

When you use the `REVIEW` command with the name of a collection, as in the following example, DEC/Test Manager automatically displays a summary of the specified collection. If you specify the `REVIEW` command without a collection name, DEC/Test Manager prompts you for one. The following example shows you how to invoke the Review subsystem for the `MSGTEST` collection.

```
DTM> REVIEW MSGTEST
Collection MSGTEST with 4 tests was created on 20-MAY-1988 10:46:34 by the command:
  CREATE COLLECTION MSGTEST MSGTEST/GROUP "Creating collection MSGTEST"
  Last Review Date = 23-MAY-1988 10:04:45
  Success count = 0
  Unsuccessful count = 0
  New test count = 4
  Updated test count = 0
  Comparisons aborted = 0
  Test not run count = 0

Result Description MESSAGE_1      Comparison Status : New Test

DTM_REVIEW>
```

To exit from the Review subsystem, type **EXIT** at the **DTM\_REVIEW>** prompt or press **CTRL/Z**. Control returns to the DCL level if you invoked the Review subsystem from the DCL prompt (**\$**); control returns to the DEC/Test Manager level if you invoked the Review subsystem from the **DTM>** prompt.

---

### 5.2.1.2 Primary and Read-only Reviewers

The **primary reviewer** can insert tests into groups, and update benchmarks in a DECwindows environment. In a DECwindows environment, test results are accessed using the view windows; being a primary reviewer in the DECwindows environment is necessary only when you insert tests into groups or update benchmarks. See Chapter 2 for information about using DEC/Test Manager views.

Only one person at a time can be the primary reviewer of a collection. If you try to access a collection as a primary reviewer while it is already being reviewed by another primary reviewer, you receive an error message. You are designated a primary reviewer of a collection if you enter the **REVIEW** command without the **/READ\_ONLY** qualifier.

You are designated a **read-only reviewer** of a collection by entering the **REVIEW** command with the **/READ\_ONLY** qualifier. You can peruse the result descriptions (see Section 5.1.1) and print files, but you cannot make any changes to the result descriptions. Read-only reviewers cannot update benchmarks or insert tests into groups. DEC/Test Manager allows multiple read-only reviewers.

As a read-only reviewer, you may obtain inaccurate information if you enter a **SHOW** command when the primary reviewer is updating benchmark files. Under the same circumstances, the Collection Summary Information also may be incorrect and files queued for printing may disappear.

As a read-only reviewer, you can use the **PRINT/NOW** command to avoid having files you select for printing disappear. See the **PRINT** command in the Command Dictionary section for more information.

---

### 5.2.1.3 Canceling Review Subsystem Commands

Press CTRL/C to cancel a transaction while it is being processed and to return control to the DTM\_REVIEW> prompt.

If you press CTRL/C during a wildcard transaction that updates the library DEC/Test Manager finishes updating the file it was processing at the time you pressed CTRL/C before it returns to the DTM\_REVIEW> prompt.

If you press CTRL/C at the DTM\_REVIEW> prompt, DEC/Test Manager terminates the Review session as if you had entered the EXIT command with the /NOPRINT and /NOINSERT qualifiers.

---

### 5.2.1.4 Locating Test Results in the Review Subsystem

When you enter the Review subsystem, the first result description of a collection is set as the current location. Test results are arranged sequentially by result description name.

To place a result description in the current review location, use one of the Review subsystem commands from Table 5-3, or press the RETURN key to move to the next result description.

**Table 5-3: Locating Test Results**

---

<b>Command</b>	<b>Description</b>
FIRST	Moves you to the first result description in the collection.
LAST	Moves you to the last result description in the collection.
NEXT	Moves you to the next result description in the collection. You can use the comparison status to move to the next test with the specified comparison status. For example, to move to the next unsuccessful test in a collection, specify NEXT/UNSUCCESSFUL. You can also move forward a specified number of tests in a collection; for example, NEXT 3 moves you forward three tests in the collection. Further, you can combine the comparison status and the count parameters to move forward to specific tests.

---

(continued on next page)

**Table 5-3 (Cont.): Locating Test Results**

<b>Command</b>	<b>Description</b>
<b>BACK</b>	Moves you to the previous result description in the collection. You can use the comparison status to move to the previous test with the specified comparison status. For example, to move to the previous unsuccessful test in a collection, specify <b>BACK/UNSUCCESSFUL</b> . You can also move backward a specified number of tests in a collection; for example, <b>BACK 3</b> moves you backward three tests in the collection. Further, you can combine the comparison status and the count parameters to move backward to specific tests.
<b>SELECT</b>	Moves you to a specific result description that you specify by its result description name. For example:  <code>DTM_REVIEW&gt; SELECT INFOMSGTEST</code>

You can use the comparison statuses to locate a set of result descriptions. For example, you can show all of the unsuccessful tests in a collection as shown in the following example:

```
DTM_REVIEW> SHOW/UNSUCCESSFUL
Result Description MSGTEST   Comparison Status : Unsuccessful

    Benchmark File is DUA0:[USER01.DTMLIB]MSGTEST.BMK
    Result file is present
    Difference file is present

Result Description INFOMSGTEST   Comparison Status : Unsuccessful

    Benchmark File is DUA0:[USER01.DTMLIB]INFOMSGTEST.BMK
    Result file is present
    Difference file is present

DTM_REVIEW>
```

For more information about the comparison status qualifiers, see the Review subsystem **SHOW** or **PRINT** commands in the Command Dictionary section.

### **Locating Test Results with DECwindows**

Depending on the view you want, you can expand a view to access the information you want by performing the following steps:

1. Ensure you have a collection view displayed.
2. Double click on the collection you want to review.

---

### 5.2.1.5 Using the Review Subsystem Keypads

The Review subsystem has several default keypads that you can use to issue commands in the command-line interface. The keypads are intended for use in the terminal environment. You can redefine these keypads to create custom keypads. Chapter 6 shows how to redefine the keypad keys for any of the default keypads. This section describes the following default keypads:

- Review subsystem
- SHOW/BENCHMARK and SHOW/RESULT
- SHOW/DIFFERENCES

In the following illustrations, the white area of a key indicates that you can issue the command by pressing the corresponding key. The shaded area of a key indicates that you can issue the command by pressing PF1 then the corresponding key.

For example, to display a benchmark file using the Review subsystem keypad (Figure 5–1), press KP9. To print a benchmark file, press PF1, then KP9.

#### NOTE

The corresponding number keys on the keyboard give you the same movement as pressing the numbers on the keypad.

Within the Review subsystem, pressing CTRL/H displays HELP, pressing CTRL/W refreshes the screen, and pressing CTRL/Z terminates display of the result, benchmark, or difference file and returns control to the Review subsystem.

Figure 5–1 shows the default Review subsystem keypad, which has most of the keypad keys defined.



**Figure 5–1: Review Subsystem Default Keypad**



ZK-4749-GE

Table 5–4 describes the key definitions in Figure 5–1.

**Table 5–4: Key Definitions for the Review Subsystem Keypad**

Key Sequence	Key Definition
KP0	Displays the next test in a collection.
PF1-KP0	Displays the previous test in a collection.
KP1	Displays the next unsuccessful test in a collection.

(continued on next page)

**Table 5–4 (Cont.): Key Definitions for the Review Subsystem Keypad**

<b>Key Sequence</b>	<b>Key Definition</b>
PF1-KP1	Displays the previous unsuccessful test in a collection.
KP2	Displays the next new test in a collection.
PF1-KP2	Displays the previous new test in a collection.
KP3	Displays the next updated test in a collection.
PF1-KP3	Displays the previous updated test in a collection.
KP4	Queues the file for immediate printing.
KP6	Updates the current test in a collection.
KP7	Issues the SHOW command with the /DIFFERENCES qualifier and displays the differences.
PF1-KP7	Prints the difference file.
KP8	Issues the SHOW/RESULTS command and displays the result file.
PF1-KP8	Prints the result file.
KP9	Issues the SHOW/BENCHMARK command and displays the benchmark file.
PF1-KP9	Prints the benchmark file.
period ( KP. )	Displays the first screen.
PF1-period ( KP. )	Displays the last screen.
comma ( KP, )	Issues the SPAWN command causing you to temporarily exit from DEC/Test Manager.
PF1-comma ( KP, )	Issues the ATTACH command to attach you to the parent process (return to the DEC/Test Manager Review subsystem).
Dash ( KP- )	Displays the collection summary.
PF2	Toggles the HELP display. If the HELP display is on the screen, pressing PF2 removes it. If the HELP display is not on the screen, pressing PF2 displays it.

Figure 5–2 shows the default SHOW/BENCHMARK, SHOW/RESULT, and DISPLAY/BENCHMARK keypad. This keypad is enabled under the following circumstances:

- When you press either KP8 or KP9 on the default Review subsystem keypad (see Figure 5–1)
- When you issue the SHOW command with the /BENCHMARK or /RESULT qualifier from the Review subsystem command line

- When you specify the DISPLAY/BENCHMARK command from the DEC/Test Manager command line

**Figure 5-2: Review Subsystem SHOW/RESULT, SHOW/BENCHMARK, and DISPLAY/BENCHMARK Keypad**

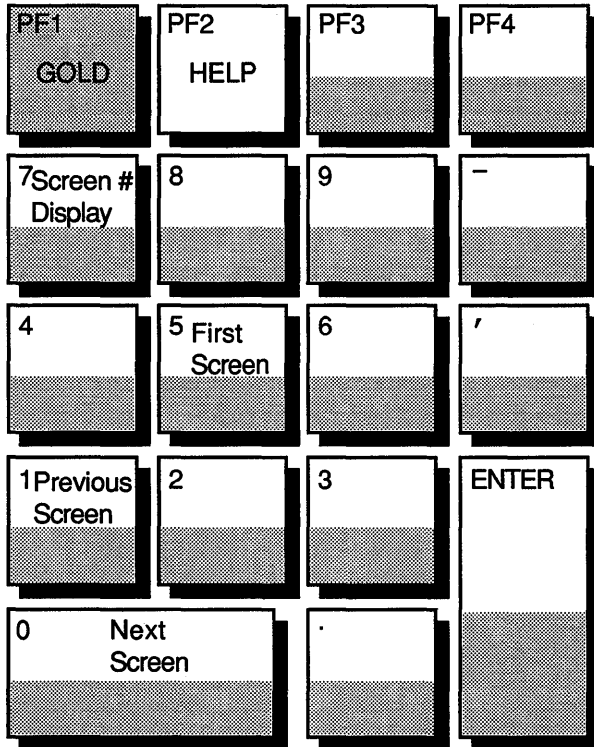


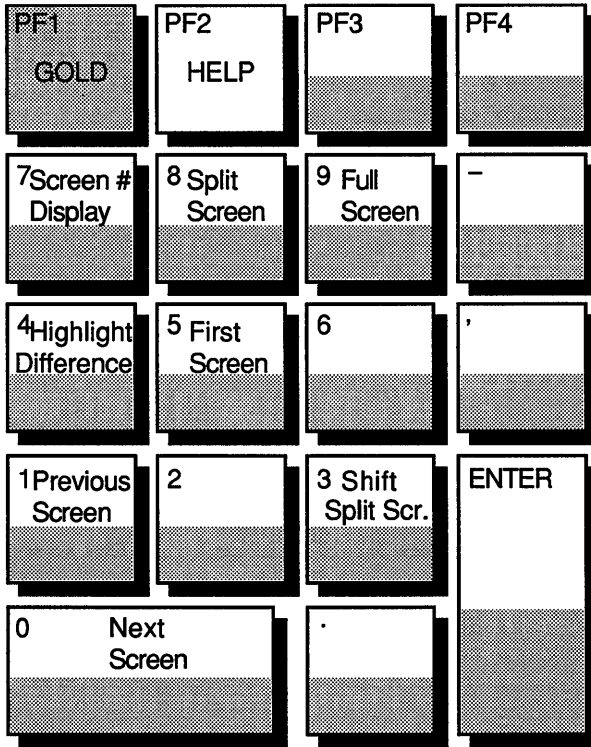
Table 5-5 describes the key definitions in Figure 5-2.

**Table 5-5: Key Definitions for the SHOW/RESULT, SHOW/BENCHMARK, and DISPLAY/BENCHMARK Keypad**

<b>Key Sequence</b>	<b>Key Definition</b>
KP0	Displays the next screen.
KP1	Displays the previous screen.
KP5	Displays the first screen in the file.
KP7	Toggles the screen number display. If the screen number display is on the screen, pressing KP7 removes it. If the screen number display is not on the screen, pressing KP7 displays it.
PF2	Toggles the HELP display. If the HELP display is on the screen, pressing PF2 removes it. If the HELP display is not on the screen, pressing PF2 displays it.

Figure 5-3 shows the SHOW/DIFFERENCES keypad. When you press KP7 on the Review keypad, or issue the SHOW command with the /DIFFERENCES qualifier from the Review subsystem command line, the default Review subsystem SHOW/DIFFERENCES keypad becomes available.

**Figure 5–3: Review Subsystem SHOW/DIFFERENCES Default Keypad**



ZK-4751-GE

Table 5–6 describes the key definitions in Figure 5–3.

**Table 5–6: Key Definitions for the SHOW/DIFFERENCES Keypad**

Key Sequence	Key Definition
KP0	Displays the next screen.
KP1	Displays the previous screen.

(continued on next page)

**Table 5–6 (Cont.): Key Definitions for the SHOW/DIFFERENCES Keypad**

<b>Key Sequence</b>	<b>Key Definition</b>
KP3	Shifts split-screen mode. If the top (or bottom) half of the result and benchmark files is displayed, pressing KP3 displays the other half of the two screens. If the right (or left) half of the result and benchmark screens is displayed, pressing KP3 displays the other half of the two screens.
KP4	Changes highlighting of differences for screens that have not been displayed. Differences are highlighted in bold reverse video. You can change this so that differences are underlined. Pressing KP4 changes highlighting for the screens that have not been displayed. After a screen is displayed, you cannot change the way its differences are highlighted.
KP5	Displays the first screen in the file.
KP7	Toggles the screen number. If the screen number is on the screen pressing KP7 removes it. If the screen number is not displayed, pressing KP7 displays it.
KP8	Toggles split-screen mode. If you are in full screen mode, pressing KP8 puts you in split-screen mode. If you are in split-screen mode, pressing KP8 switches you between horizontal split-screen mode and vertical split-screen mode.
KP9	Toggles full-screen mode. If you are in split-screen mode, pressing KP9 puts you in full-screen mode. If you are in full-screen mode, pressing KP9 switches you between displaying full screens from the result and benchmark files.
PF2	Toggles the HELP display. If HELP is displayed, pressing PF2 removes it. If HELP is not displayed, pressing PF2 displays it.

## 5.2.2 Displaying Test Results

This section describes how to display test results in the terminal environment. See Chapter 2 for information about displaying tests in a DECwindows environment.

The Review subsystem SHOW command displays information about result descriptions and displays the output files associated with result descriptions to the display device (SYS\$OUTPUT).

You can specify a result description expression with the **SHOW** command, which identifies the result descriptions about which information is to be displayed. If you specify no result description, DEC/Test Manager displays information about the current result description.

The following qualifiers enable you to display specified output files individually or in groups:

- /BENCHMARK**
- /DIFFERENCE**
- /RESULT**

If you do not include an output file qualifier, DEC/Test Manager displays information about the specified result descriptions rather than an output file.

The following qualifiers enable you to display groups of result descriptions that have the same comparison status.

- /COMPARISON\_ABORTED**
- /NEW**
- /NOT\_RUN**
- /SUCCESSFUL**
- /UNSUCCESSFUL**
- /UPDATED**

The **SHOW** command also takes the following qualifiers:

- /FILES**
- /OUTPUT**
- /SUMMARY**

The **/FILES** qualifier displays the comparison status for result descriptions and states whether each output file exists. You cannot use the **/FILES** qualifier with the **/SUMMARY** qualifier.

The **/OUTPUT** qualifier directs the output to a specified file.

The **/SUMMARY** qualifier displays the collection summary information; the information displayed when you invoke the Review subsystem. You cannot use the **/SUMMARY** qualifier with the comparison status or the **/FILES** qualifier.

When you enter the **SHOW** command to display an output file for any type of test, DEC/Test Manager automatically recognizes the test as a noninteractive or interactive terminal test and displays it accordingly. The following example shows the output for a noninteractive terminal test.

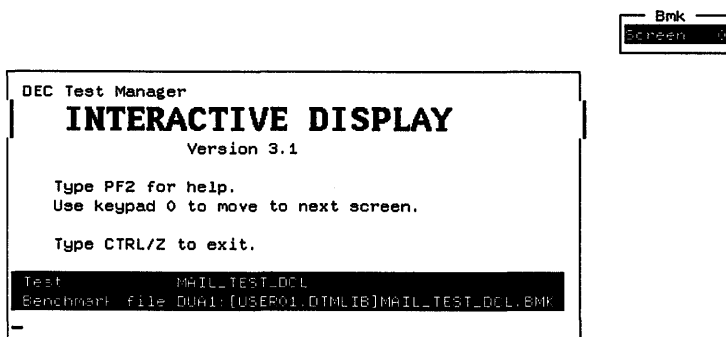
```
DTM_REVIEW> SHOW/BENCHMARK
Benchmark File DUA0:[USER01.DTMLIB]MSGTEST.BMK For Result Description
MSGTEST
.
.
.
DTM_REVIEW>
```

When you enter a **SHOW** command with an output file qualifier for an interactive terminal test, DEC/Test Manager displays the result or benchmark files screen by screen, record by record, or character by character (depending on comparison type).

For a comparison using the **/SCREENS** qualifier, the differences are displayed in split-screen mode; the top half of a result file screen is displayed on the top half of the screen, and the top half of the corresponding benchmark screen is displayed on the bottom half of the screen. You can also view the differences in a full-screen mode.

When you enter the **SHOW** command with the **/RESULT** or **/BENCHMARK** qualifiers, an initial banner screen (Screen 0) is displayed. Subsequent screens are numbered for easy referencing. Pressing **RETURN** accesses the next screen. Figure 5-4 shows Screen 0 for a benchmark file. Screen 0 for a result file is similar.

**Figure 5-4: DEC/Test Manager Benchmark File Screen 0**



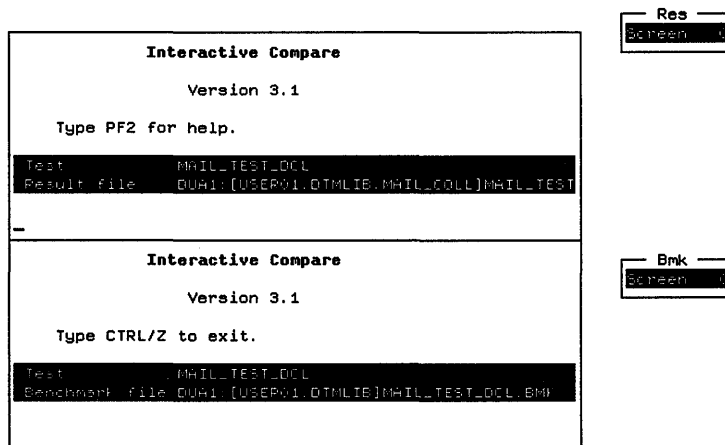
When you display a result or benchmark file, the specified file is displayed screen by screen. The screen number, initially on the top right corner of the screen, shows the number of the screen and the type of file currently displayed.



To view other screens in the result and benchmark files, you must use the default keypad for the SHOW/RESULT and SHOW/BENCHMARK screens. Section 5.2.1.5 describes the SHOW/RESULT or SHOW/BENCHMARK keypad keys and their associated functions.

When you enter the SHOW command with the /DIFFERENCE qualifier, an initial banner screen (Screen 0) is displayed, as shown in Figure 5-5.

**Figure 5-5: DEC/Test Manager Difference File Screen 0**



When you display a difference file, you are in horizontal split-screen mode by default; the terminal screen is divided horizontally into two windows. The top window displays the top half of a screen from the result file, and the bottom window displays the top half of the corresponding screen from the benchmark file.

To view other screens in the result or benchmark file, you must use the default keypad for the SHOW/DIFFERENCES screen. Section 5.2.1.5 describes the SHOW/DIFFERENCES keypad keys and their associated functions.

---

### 5.2.3 Printing Test Results

The Review subsystem PRINT command marks the specified output files for placement in the default printer queue.

#### NOTE

The PRINT command does not apply to DECwindows files. DECwindows result, benchmark, and difference files are stored in DDIF format. They may require conversion to another format for printing.

The PRINT command does not queue the selected files for printing until after you exit from the Review subsystem with the EXIT command or by pressing CTRL/Z. If you leave the Review subsystem by pressing CTRL/C or entering the EXIT/NOPRINT/NOINSERT command, the selected files are not printed.

If you do not include an output file qualifier, DEC/Test Manager prints the result file associated with the current result description.

In addition to the output file and comparison status qualifiers, and the standard print qualifiers, the PRINT command also takes the following qualifiers:

`/[NO]LOG`  
`/NOW`  
`/SELECTED`

The `/[NO]LOG` qualifier controls whether DEC/Test Manager displays informational and success messages on the display device. The `/LOG` qualifier is the default.

The `/NOW` qualifier places all specified files immediately in the print queue.

The `/SELECTED` qualifier concatenates all files already selected for printing with the currently specified files and immediately places them in the print queue.

#### NOTE

If you select a result file for printing and subsequently update the benchmark file for that test before the result file has been queued for printing, the result file is deleted and is no longer printable.

---

## 5.3 Working with Test Results

This section summarizes the procedures for working with test results in the Review subsystem. You must be the primary reviewer of a collection before you can update any benchmark files.

While in the Review subsystem, you can use the `UPDATE` command to create or replace a benchmark file for a test. The `UPDATE` command makes the new benchmark file out of the current result file for the specified result descriptions. DEC/Test Manager also provides DECwindows menu access for updating functions.

If a benchmark file already exists in the DEC/Test Manager library, it is deleted when you enter the `UPDATE` command. If a benchmark file already exists but is outside the DEC/Test Manager library, DEC/Test Manager informs you of this and replaces it as the benchmark file; it does not delete the old file.

Section 5.3.1 shows how to use Review subsystem commands to replace a test's benchmark file; Section 5.3.2 shows how to use Review subsystem commands to create a benchmark file for a new test.

See Chapter 2 for information on updating a benchmark file in a DECwindows environment.

---

### 5.3.1 Updating an Existing Benchmark File

You might want to update an existing benchmark file if you have changed an application in a way that would change the expected results for a test.

Example 5-1 assumes that you previously examined the result file or difference file and determined that the result file should replace the old benchmark file. First, locate the result description for the test by entering the `SELECT` command in the Review subsystem. Then, enter the `UPDATE` command.

## Example 5-1: Updating a Benchmark File

---

```
DTM_REVIEW> SELECT MSGTEST
Result Description MSGTEST      Comparison Status : Unsuccessful

DTM_REVIEW> UPDATE
%DTM_I_UPDATED, the benchmark for test MSGTEST has been updated

DTM_REVIEW>
```

---

The result file is renamed as the benchmark file and the reference to the former benchmark file is removed. If the former benchmark file is in the DEC/Test Manager library, it is deleted.

When you update a benchmark file that is stored in a VAX DEC/Code Management System (CMS) library, DEC/Test Manager reserves the existing benchmark element in the CMS library and replaces it with the result file from the current test run. If you specified a CMS generation for the existing benchmark file with the CREATE COLLECTION/CLASS command, DEC/Test Manager inserts the updated benchmark file as the new generation in the specified CMS class.

If you update the benchmark file for an interactive terminal test, the file containing benchmark screens is also updated. If you update the benchmark file for a DECwindows test, any masks in the existing benchmark are transferred to the new benchmark.

---

### 5.3.2 Creating a Benchmark File for a New Test

You might want to create a benchmark file for a new test. This section shows you how to do it in the Review Subsystem.

Use the Review subsystem to create a benchmark file for a new test in a terminal environment. The following tasks are accomplished in Example 5-2:

- Invoking the Review subsystem to review a collection
- Locating the result description for a test
- Displaying a result file
- Creating a benchmark file for the test by renaming the result file as the benchmark file (using UPDATE)
- Showing the collection summary
- Printing the newly created benchmark file

The reverse-print numbers refer to the command line explanations in the list that follows the example. Note that when you omit the result description expression, the SHOW, UPDATE, and PRINT commands all refer to the current result description.

### Example 5-2: Creating a Benchmark File

---

- ① DTM> REVIEW MAIL\_COLL  
Collection MAIL\_COLL with 1 test was created on 31-OCT-1988 07:19:16  
by the command:  
    CREATE COLLECTION MAIL\_COLL MAIL\_TEST "Creating the MAIL test  
    collection"  
    Last Review Status = not previously reviewed  
    Success count = 0  
    Unsuccessful count = 0  
    New test count = 1  
    Updated test count = 0  
    Comparisons aborted = 0  
    Test not run count = 0  
  
Result Description MAIL\_TEST           Comparison status : New test
  - ② DTM\_REVIEW> SHOW/RESULT  
    .  
    .  
    .
  - ③ DTM\_REVIEW> UPDATE  
%DTM-I-UPDATED, the benchmark file for test MAIL\_TEST has been updated
  - ④ DTM\_REVIEW> SHOW/SUMMARY  
Collection MAIL\_COLL with 1 test was created on 31-OCT-1988 07:19:16  
by the command:  
    CREATE COLLECTION MAIL\_COLL MAIL\_TEST "Creating the MAIL test  
    collection"  
    Last Review Date = 02-NOV-1988 08:19:20  
    Success count = 0  
    Unsuccessful count = 0  
    New test count = 0  
    Updated test count = 1  
    Comparisons aborted = 0  
    Test not run count = 0
  - ⑤ DTM\_REVIEW> PRINT/BENCHMARK  
%DTM-S-PRINT, file DUA0:[USER01.DTMLIB]MAIL\_TEST.BMK of test MAIL\_TEST  
selected for printing
  - ⑥ DTM\_REVIEW> EXIT  
%DTM-S-PRINTQD, print job has been sent to the print queue  
-DTM-I-TEXT, Job MAIL\_TEST(queue SYS\$PRINT, entry 710)started on SYS\$PRINT  
%DTM-EXIT, leaving Review subsystem  
DTM>
-

- ❶ Invoke the Review subsystem to review the MAIL\_COLL collection. Note that the collection was not reviewed previously and that it contains one new test, MAIL\_TEST.
- ❷ Display a result file to determine whether the results are correct. It is unnecessary to specify the result description name, because MAIL\_TEST is the current result description. If the results are correct, you can then make this file the benchmark file for the test. Because MAIL\_TEST is a noninteractive test, DEC/Test Manager displays the result file on the screen.
- ❸ Use the UPDATE command to create a benchmark file for the test by renaming the result file as the benchmark file. This command creates a benchmark file for MAIL\_TEST from its result file, deletes the result file, and changes the comparison status of the test from new to updated. DEC/Test Manager stores the benchmark file in the DEC/Test Manager library and uses it as the benchmark file for MAIL\_TEST when the test is run in future collections.

If you are creating the benchmark file for an interactive test, the file containing benchmark screens (test-name.BMK\_SCREEN) is also created. These files are printable for interactive terminal tests.

- ❹ Show the collection summary. Note that the New test count field value is 0 and that the Updated test count field value is 1.
- ❺ Print the newly created benchmark file with the PRINT command.
- ❻ Exit from the Review subsystem.

#### **NOTE**

When you create a new benchmark file in a VAX DEC/Code Management System (CMS) library, DEC/Test Manager creates a new benchmark element in the specified CMS library. If you specify a CMS class with the CREATE COLLECTION/CLASS command, DEC/Test Manager inserts the new benchmark file as a generation in the class. See Chapter 7 for more information about using VAX DEC/Code Management System with DEC/Test Manager.

---

### 5.3.3 Reviewing Partially Run Collections

A partially run collection may occur for one of the following reasons:

- You stopped a collection by typing the **STOP** command if the collection was executing in batch, or by pressing **CTRL/C** or **CTRL/Y** if the collection was running interactively.
- Your collection terminated abnormally during execution.
- You used the **DCL DELETE** command with the **/ENTRY** qualifier to stop a collection of tests running in batch.
- Someone stopped the process executing the collection.

Issuing the **STOP** command is the recommended way to terminate a collection executing in batch. Pressing **CTRL/C** is the recommended way to stop a collection that is running interactively. Taking either of these actions causes **DEC/Test Manager** to clean up the **DEC/Test Manager** library, after which you must enter the **COMPARE** command to compare the partially run collection before reviewing it.

If you do not use the **STOP** command or press **CTRL/C** to stop a collection, errors may occur and you will not be able to compare or review the collection. Before using the collection, you must enter the **VERIFY/RECOVER** command to clean up the library, correct the errors, and mark the tests that did not run. Then, you must enter the **COMPARE** command to compare the partially run collection. Finally, you must enter the **REVIEW** command to initiate a Review session for the partially run collection.

#### **NOTE**

For partially run or partially compared collections, all tests that did not run are marked with the not run comparison status. In both cases, you must perform a comparison before reviewing the collection.

If you review the collection before comparing it, **DEC/Test Manager** will not allow you to compare the collection later. If the collection is stopped after some tests have been compared, those tests will retain the correct comparison status.

Reviewing a partially run collection is especially important if the collection is large. Following the instructions just described, you should prepare the partially run collection for review. Then, from the Review subsystem, examine the tests that ran and use the `INSERT/NOT_RUN` command to create a group containing all the tests that did not run. This group can then be included in a collection, executed, and reviewed. Creating a group while reviewing a collection is described in Chapter 6.



# Tailoring Your Test System

---

This chapter describes features of DEC/Test Manager that you can use to tailor your DEC/Test Manager test system to suit your testing needs; it provides information on the following topics:

- Using prologue and epilogue files
- Creating and using groups
- Using variables
- Using filters
- Using masks
- Defining keypad keys
- Using command files
- Creating initialization files
- Spawning and attaching processes

---

## 6.1 Using Prologue and Epilogue Files

**Prologue** and **epilogue** files are command files that enable you to control the environment in which DEC/Test Manager runs your tests. You store prologues and epilogues outside the DEC/Test Manager library, in VMS directories or in VAX DEC/Code Management System (CMS) libraries.

You use prologues to set up test conditions before executing a test in a collection or before executing the collection as a whole. You use epilogues to clean up or filter files after executing a test in a collection or after executing the collection as a whole.

Table 6–1 describes the various prologue and epilogue files.

**Table 6–1: Prologue and Epilogue Files**

<b>File</b>	<b>Description</b>
Test prologue	Associated with a specific test description, it is executed before the test template file is executed.
Test epilogue	Associated with a specific test description, it is executed after the test template executes and DTM\$RESULT is defined, but before the DEC/Test Manager-provided filters are run.
Collection prologue	Associated with a collection, it is executed whenever the collection is executed after the global variables and DTM\$COLLECTION_NAME are defined, but before any tests are executed.
Collection epilogue	Associated with a collection, it is executed after the collection is executed and is the last file executed.

### 6.1.1 Test Prologue and Epilogue Files

You create test prologue and epilogue files using a text editor and associate them with tests using one of the following commands with the /PROLOGUE and /EPILOGUE qualifiers:

- CREATE TEST\_DESCRIPTION
- MODIFY TEST\_DESCRIPTION

In a DECwindows environment, you specify test prologue and test epilogue files in text entry fields on the Create Test... or Modify Test... dialog box.

Test prologue and epilogue files are associated with a specific test description and are executed whenever the test is executed.

A prologue file can set up an environment for the test template file. For example, a test prologue file can define local variables, FETCH elements from a CMS library, or set default values.

An epilogue file can clean up the environment after the template file is run. Test epilogue files can also remove run-dependent data from a test's result file. By running a test once and checking for run-dependent data in the result file, you can determine the need to filter run-dependent data with an epilogue file. See Section 6.3.3.3 and Section 6.4 for more information.

You can disassociate a prologue or epilogue file from a test description by specifying the `MODIFY TEST_DESCRIPTION` command with the `/NOPROLOGUE` or `/NOEPILOGUE` qualifiers. This does not delete the prologue or epilogue file.

Example 1–1 in Chapter 1 shows the disabling and enabling of broadcast messages as part of a recorded test. However, it is possible that broadcast messages could appear in the recording before the `SET BROADCAST=NONE` command was entered, or after the `SET BROADCAST=ALL` command was entered and before the test recording was terminated. By placing these commands into test prologue and epilogue files, you eliminate that possibility. The following sections show you how to place these commands into test prologue and epilogue files.

Example 6–1 shows a simple test prologue file to disable broadcast messages before test recording begins.

### Example 6–1: Sample Test Prologue File

---

```
$! DTMNOBROADCAST.COM
$!
$! Disable broadcast messages before recording.
$!
$ SET BROADCAST=NONE
$!
```

---

To establish this test prologue file for the `MAIL_TEST` test, specify the following command:

```
DTM> MODIFY TEST_DESCRIPTION MAIL_TEST -
_DTM> /PROLOGUE=DUA0:[USER01.PROLIB]DTMNOBROADCAST.COM
_Remark: Adding the prologue to disable broadcast messages
%DTM-S-MODIFIED, test description MAIL_TEST modified
DTM>
```

Example 6–2 shows a simple test epilogue file to enable broadcast messages after test recording concludes.

## Example 6-2: Sample Test Epilogue File

---

```
$! RESETBROADCAST.COM
$!
$! Re-enable broadcast messages after recording.
$!
$ SET BROADCAST=ALL
$!
```

---

To establish this test epilogue file for the MAIL\_TEST test, specify the following command:

```
DTM> MODIFY TEST_DESCRIPTION MAIL_TEST -
_DTM> /EPILOGUE=DUA0:[USER01.EPILIB]RESETBROADCAST.COM
_Remark: Adding the epilogue to re-enable broadcast messages
%DTM-S-MODIFIED, test description MAIL_TEST modified
DTM>
```

---

### 6.1.2 Collection Prologues and Epilogues

You create collection prologue and epilogue files using a text editor and set up the default collection command file specifications for the DEC/Test Manager library using one of the following commands:

- SET PROLOGUE
- SET EPILOGUE

To set collection prologue and epilogue files in a DECwindows environment, perform the following steps:

1. Pull down the Library menu.
2. Choose the Create... or Modify... menu item.
3. Fill in the Prologue and Epilogue text entry fields.

When you associate prologue and epilogue files with a library using the SET PROLOGUE and SET EPILOGUE commands, those files become the default prologue and epilogue files for any collections you create. The default prologue and epilogue files are invoked whenever you execute one of those collections.

To cancel the default files, specify the SET NOPROLOGUE and SET NOEPILOGUE commands.

To associate different prologue and epilogue files when creating subsequent collections, specify new files with the CREATE COLLECTION command and the /PROLOGUE and /EPILOGUE qualifiers.

In a DECwindows environment, you specify collection prologue and collection epilogue files in text entry fields on the Create Collection... dialog box.

To create a collection that does not use the default collection prologue and epilogue files, specify the CREATE COLLECTION command with the /NOPROLOGUE and /NOEPILOGUE qualifiers.

Example 6-3 shows a sample collection prologue file that does two things:

1. Tests the DEC/Test Manager variable USE\_PCA to determine whether to process the VAX Performance and Coverage Analyzer (PCA) Collector initialization file.
2. Defines the Collector initialization file according to the DEC/Test Manager variable USE\_PCA\_INIT\_FILE. Running the PCA Collector during a test is especially useful for determining which code paths are being exercised by the tests themselves. See the *Guide to VAX Performance and Coverage Analyzer* for information about PCA. See *Using VAXset* for information about using DEC/Test Manager with PCA.

### Example 6-3: Sample Collection Prologue File

---

```
$!           --- COLLECTION_PROLOGUE.COM ---
$! Collection prologue file for running the Collector in batch mode
$!
$ SET VERIFY
$!
$ TRANSLIT:==$'F$LOGICAL("TRANSLIT")'
$!
$! Test DTM variable to determine whether or not to run PCA prologue
$!
$ IF USE_PCA .EQS. "FALSE" THEN EXIT
$!
$! Define the Collector initialization file
$!
$ DEFINE PCAC$INIT USE_PCA_INIT_FILE
$!
$! End of collection prologue file
```

---

To establish this collection prologue file as the default prologue file for subsequently created test collections, specify the following command:

```
DTM> SET PROLOGUE DUA0:[USER01.DTM_CMSLIB]COLLECTION_PROLOGUE.COM
```

In this example, the prologue file exists in the CMS library.

Example 6-4 shows an epilogue file that sends the results of the tests in a collection to the project leader through the VMS Mail Utility (MAIL). The collection name is identified by the DEC/Test Manager logical name DTM\$COLLECTION\_NAME; the project leader is identified with the mail address, USER87.

---

#### Example 6-4: Sample Collection Epilogue File

---

```
$!          --- COLLECTION_EPILOGUE.COM ---
$! Collection epilogue file for mailing test results to USER87,
$! upon completion of the test run.
$!
$ DTM SHOW COLLECTION 'DTM$COLLECTION_NAME'/FULL-
$ /OUTPUT='DTM$COLLECTION_NAME'.REPORT
$!
$ MAIL 'DTM$COLLECTION_NAME'.REPORT/SUBJECT="Collection summary" USER87
$!
$! End of collection epilogue file
```

---

To establish this epilogue file as the default epilogue file for subsequently created test collections, specify the following command:

```
DTM> SET EPILOGUE DUA0:[USER01.DTM_CMSLIB]COLLECTION_EPILOGUE.COM
```

In this example, the epilogue file exists in the CMS library.

---

## 6.2 Grouping Tests

You can classify test descriptions by placing them into categories called **groups**. You identify each group in the library with a **group name**, which is unique in the current library. As a result, you need only specify a group name rather than a long list of individual test descriptions when referencing the tests in a group.

This section describes the DEC/Test Manager commands that perform the following actions:

- Create a group
- Change the contents of a group
- Delete a group
- Build a hierarchy of groups

Table 6–2 shows the commands that operate on groups and their corresponding functions.

**Table 6–2: Group Commands**

<b>Command</b>	<b>Function</b>
CREATE GROUP	Creates an empty group into which you can insert tests or groups.
DELETE GROUP	Deletes an existing group. A group must be empty before it can be deleted.
INSERT GROUP	Places a group inside another group.
INSERT TEST_DESCRIPTION	Places a test inside an existing group.
MODIFY GROUP	Replaces the remark associated with an existing group.
REMOVE GROUP	Removes a group from another group.
REMOVE TEST_DESCRIPTION	Removes a test from an existing group.
SHOW GROUP	Lists the group's name, its contents, and its creation remark.

---

## 6.2.1 Organizing Tests into Groups

After you create a group name with the `CREATE GROUP` command, you can insert one or more tests by listing the test names with the `INSERT TEST_DESCRIPTION` command. You can use wildcards when you specify test names to be inserted into a group. Tests are associated together by group name only; they are not relocated or copied.

Example 6–5 shows how to create several groups in a terminal environment; it uses `CREATE GROUP`, `INSERT TEST_DESCRIPTION`, and `INSERT GROUP` commands.

## Example 6–5: Creating Groups

---

```
❶ DTM> CREATE GROUP BOUNDARIES "Creating Group BOUNDARIES"
%DTM-S-CREATED, group BOUNDARIES created
DTM> CREATE GROUP LMARGIN "Creating Group LMARGIN"
%DTM-S-CREATED, group LMARGIN created
DTM> CREATE GROUP RMARGIN "Creating Group RMARGIN"
%DTM-S-CREATED, group RMARGIN created
DTM> CREATE GROUP MARGINS "Creating Group MARGINS"
%DTM-S-CREATED, group MARGINS created

❷ DTM> INSERT TEST_DESCRIPTION LMTEST1,LMTEST2,LMTEST3 LMARGIN
_Remark: Grouping the left margin tests
%DTM-I-INSERTED, test description LMTEST1 inserted into group LMARGIN
%DTM-I-INSERTED, test description LMTEST2 inserted into group LMARGIN
%DTM-I-INSERTED, test description LMTEST3 inserted into group LMARGIN
%DTM-S-INSERTIONS, 3 insertions completed

❸ DTM> INSERT TEST_DESCRIPTION RMTEST1,RMTEST2,RMTEST3,RMTEST4 RMARGIN
_Remark: Grouping the right margin tests
%DTM-I-INSERTED, test description RMTEST1 inserted into group RMARGIN
%DTM-I-INSERTED, test description RMTEST2 inserted into group RMARGIN
%DTM-I-INSERTED, test description RMTEST3 inserted into group RMARGIN
%DTM-I-INSERTED, test description RMTEST4 inserted into group RMARGIN
%DTM-S-INSERTIONS, 4 insertions completed

❹ DTM> INSERT GROUP LMARGIN,RMARGIN MARGINS
_Remark: Grouping the margin tests together under MARGINS
%DTM-I-INSERTED, group LMARGIN inserted into group MARGINS
%DTM-I-INSERTED, group RMARGIN inserted into group MARGINS
%DTM-S-INSERTIONS, 2 insertions completed

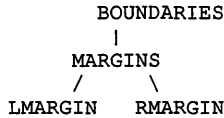
❺ DTM> INSERT GROUP MARGINS BOUNDARIES
_Remark: Grouping the margin groups into a boundaries group
%DTM-I-INSERTED, group MARGINS inserted into group BOUNDARIES
DTM>
```

- 
- ❶ Create four empty groups: BOUNDARIES, LMARGIN, RMARGIN, and MARGINS.
  - ❷ Insert three left margin tests into the LMARGIN group.
  - ❸ Insert four right margin tests into the RMARGIN group.
  - ❹ Insert the two groups LMARGIN and RMARGIN into the group MARGINS.
  - ❺ Insert the group MARGINS into the group BOUNDARIES.

When you insert a group into another group, you create a **group hierarchy**. Figure 6–1 shows the BOUNDARIES groups hierarchy.



**Figure 6–1: Sample Group Hierarchy**



## 6.2.2 Displaying a Group Structure

Use the **SHOW GROUP** command to display the structure of groups for the current library by specifying the **SHOW GROUP** command. You can display different amounts of information about the groups in a library with the **SHOW GROUP** qualifiers (see the Command Dictionary section for more information). The following example shows the output for the **SHOW GROUP** command with the **/FULL** qualifier for the project library:

```
DTM> SHOW GROUP/FULL

Groups in DEC/Test Manager Library DUA0:[USER01.PROJECT]

BOUNDARIES    "Creating Group BOUNDARIES"
  MARGINS/Group
    LMARGIN/Group
      LMTEST1
      LMTEST2
      LMTEST3
    RMARGIN/Group
      RMTEST1
      RMTEST2
      RMTEST3
      RMTEST4

LMARGIN       "Creating Group LMARGIN"
  LMTEST1
  LMTEST2
  LMTEST3

MARGINS       "Creating Group MARGINS"
  LMARGIN/Group
    LMTEST1
    LMTEST2
    LMTEST3
  RMARGIN/Group
    RMTEST1
    RMTEST2
    RMTEST3
    RMTEST4
```

```
RMARGIN          "Creating Group RMARGIN"
  RMTEST1
  RMTEST2
  RMTEST3
  RMTEST4
DTM>
```

---

## 6.2.3 Removing Tests and Subgroups from Groups

Use the REMOVE TEST\_DESCRIPTION to disassociate tests from a group. Use the REMOVE GROUP commands to disassociate groups from a group. These commands reverse the actions of the INSERT TEST\_DESCRIPTION and INSERT GROUP commands. The remove commands do not delete any tests or groups.

The following example removes a single test from the LMARGIN group:

```
DTM> REMOVE TEST_DESCRIPTION LMTEST2 LMARGIN
_Remark: Removing test 2 from LMARGIN
Confirm removal of test description LMTEST2 from group LMARGIN [Y/N] (N): Y
%DTM-I-REMOVED, test description LMTEST2 removed from group LMARGIN
DTM>
```

The following example removes the RMARGIN subgroup from the MARGINS group:

```
DTM> REMOVE GROUP RMARGIN MARGINS "Removing RMARGIN from MARGINS"
Confirm removal of group RMARGIN from group MARGINS [Y/N] (N): Y
%DTM-I-REMOVED, group RMARGIN removed from group MARGINS
DTM>
```

---

## 6.2.4 Deleting Groups

When you create a group, DEC/Test Manager continues to associate that name with a group, even if the group no longer contains tests. You can delete the associated group name by specifying the DELETE GROUP command.

Before you can delete a group, you must remove all test or group association from that group, using the REMOVE TEST\_DESCRIPTION and REMOVE GROUP commands, as described in Section 6.2.3. The following example removes all the tests from the RMARGIN group, then deletes the RMARGIN group.

```
DTM> REMOVE TEST_DESCRIPTION * RMARGIN
  _Remark: Preparing to delete group RMARGIN
Confirm removal of test description RMTEST1 from group RMARGIN [Y/N] (N): Y
%DTM-I-REMOVED, test description RMTEST1 removed from group RMARGIN
.
.
.
Confirm removal of test description RMTEST4 from group RMARGIN [Y/N] (N): Y
%DTM-I-REMOVED, test description RMTEST4 removed from group RMARGIN
%DTM-S-REMOVALS, 4 removals completed
DTM> DELETE GROUP RMARGIN "Deleting the RMARGIN group"
Confirm deletion of group RMARGIN [Y/N] (N): Y
%DTM-S-DELETED, group RMARGIN deleted
DTM>
```

---

## 6.3 Using Variables

A DEC/Test Manager **variable** is a user-defined VMS symbol or logical name that you use in DEC/Test Manager tests, prologue files, and epilogue files.

Use the **CREATE VARIABLE** command to add a variable to the DEC/Test Manager library. Use the **MODIFY VARIABLE** to change file names. For example, you might want to use a variable to replace a file name in a template file. See the Command Dictionary for more information about the **CREATE VARIABLE** and **MODIFY VARIABLE** commands and their qualifiers.

### NOTE

P1 through P8 and any variable name with the prefix **DTM\$** are reserved for exclusive use by DEC/Test Manager. You receive a warning if you attempt to create a variable with these names.

A DEC/Test Manager variable can be global or local in scope. A global variable is associated with all tests in a collection; a local variable is defined only during a specific test execution. Local variables must be associated with specific test descriptions by using the **CREATE TEST\_DESCRIPTION** or **MODIFY TEST\_DESCRIPTION** command with the **/VARIABLE** qualifier. When you create a collection, DEC/Test Manager associates all existing global variables with the collection and defines them at the start of every collection.

If you redefine a global variable or create a new global variable in the DEC/Test Manager library, use the **RECREATE** command to re-create the collection and associate it with the new global variables.

The following example shows you how to use the **CREATE VARIABLE** command to create a global variable. If you have a test that issues many **SUBMIT** commands and you do not want to print all the LOG files that the test generates, you can create a variable with variable name **SUBMIT** and give it the variable value **SUBMIT/NOPRINTER**, as shown in this example.

```
DTM> CREATE VARIABLE SUBMIT "SUBMIT/NOPRINTER"/GLOBAL/SYMBOL
_Remark: "Redefine the SUBMIT command"
```

Any test that uses the **SUBMIT** command subsequent to this command uses the new definition, as if you had entered the following command before executing the test:

```
$ SUBMIT == SUBMIT/NOPRINTER
```

### NOTE

Avoid assigning values to global variables from within template, prologue, or epilogue files. Instead, let DEC/Test Manager assign the indicated value before the template, prologue, or epilogue file is executed.

Global variable values can be overridden for an individual test that requires special handling. See Section 6.3.2 for more information.

A local variable can be accessed by a single test when you include that variable in the test description. To use a local variable, you must do the following:

1. Specify the **CREATE VARIABLE** or **MODIFY VARIABLE** command with the **/LOCAL** qualifier to define the variable as a local variable.
2. Specify the variable using the **CREATE TEST\_DESCRIPTION** or **MODIFY TEST\_DESCRIPTION** command with the **/VARIABLE** qualifier, as shown in the following example:

```
DTM> CREATE TEST_DESCRIPTION MECHANIX -
_DTM> /VARIABLE=(SBMT="SUBMIT/NOPRINTER")
```

3. Use the variable in the template file, prologue file, or epilogue file of specific test descriptions.

Wherever the **SUBMIT** variable is used in the **MECHANIX** test, DEC/Test Manager translates the variable to **SUBMIT/NOPRINTER**. Other tests are unaffected by this local definition of the **SUBMIT** variable.

You can disassociate variables from an existing test description by using the **MODIFY TEST\_DESCRIPTION** command with the **/NOVARIABLE** qualifier.

---

## 6.3.1 Modifying and Deleting Variables

You can modify one or more variable characteristics and delete variables.

To modify variable characteristics, use the **MODIFY VARIABLE** command with one or more of its qualifiers. For example, you can change the default value of a variable, as shown in the following example:

```
DTM> MODIFY VARIABLE INPUT_FILE/VALUE=INPUT.RNO
_Remark: "Replacing value of INPUT_FILE with INPUT.RNO"
%DTM-S-MODIFIED, variable INPUT_FILE modified.
```

The variable expression parameter can be a variable name, a wildcard character, a wildcard in combination with a variable name, or a list of these separated by commas.

To delete a variable from the DEC/Test Manager library, use the **DELETE VARIABLE** command, as shown in the following example:

```
DTM> DELETE VARIABLE INPUT_FILE "Deleting the INPUT_FILE variable"
Confirm deletion of variable INPUT_FILE [Y/N] (N): Y
%DTM-S-DELETED, variable INPUT_FILE deleted.
```

### NOTE

DEC/Test Manager will not delete a variable that is associated with a test description. If you attempt to delete several variables with a variable expression and one or more of them is associated with a test description, DEC/Test Manager deletes only those variables not associated with a test description. Use the **MODIFY TEST\_DESCRIPTION/NOVARIABLE** command to disassociate variables from test descriptions.

See the Command Dictionary for more information about modifying and deleting variables.

---

## 6.3.2 Overriding Variable Default Values

Most tests use a variable's default value. However, certain tests may require special handling and require special variable values. For example, you may want to use one template file to run several tests. You can do this by using a variable in the template file to override the variable's value for each test description.

Example 6–6 performs the following actions:

- Creates the variable `TEMPLDIR`, with `DUA0:[USER01.TMP]` as its value.
- Modifies the variable's value so that when it is used with the test description `MECHANIX`, its value is `DUA0:[USER01.PROJECT.TMP]`.

---

### Example 6–6: Overriding Variables

---

```
DTM> CREATE VARIABLE TEMPLDIR DUA0:[USER01.TMP]/GLOBAL
_Remark: Template Directory
%DTM-S-CREATED, logical variable TEMPLDIR created
DTM> MODIFY TEST_DESCRIPTION MECHANIX -
_DTM> /VARIABLE=(TEMPLDIR=DUA0:[USER01.PROJECT.TMP])
_Remark: "Change variable value when used in MECHANIX"
%DTM-S-MODIFIED, test description MECHANIX modified
DTM>
```

---

You can override the default value of a global variable when you create a collection. To do this, use the `CREATE COLLECTION/VARIABLE` command. The following example creates the collection `KEYTESTS` and, for this collection only, changes the value of `TEMPLDIR` to `DUA0:[USER01.TEST.TMP]`.

```
DTM> CREATE COLLECTION KEYTESTS * -
_DTM> /VARIABLE=(TEMPLDIR="DUA0:[USER01.TEST.TMP]")-
_Remark: New template directory for this collection
```

The test descriptions in collection `KEYTESTS` that are explicitly associated with the variable `TEMPLDIR` are not permanently affected by the override value. For example, `MECHANIX` in `KEYTESTS` will still have the value `DUA0:[USER01.PROJECT.TMP]`, despite the collection override value.

### NOTE

You should override variables sparingly because you are changing the variable's value from earlier test runs. This may cause the actual test output to differ from the expected test output. In addition, any changes in variable values may affect the prologue and epilogue files. As a result, you must examine the differences and result files to discover whether the actual test output is what you expected.

---

### 6.3.3 Using Variables Defined by DEC/Test Manager

DEC/Test Manager supplies built-in variables that you can use in template files, prologue files, and epilogue files. The following sections describe these built-in variables and provide examples of how to use each one.

---

#### 6.3.3.1 DTM\$COLLECTION\_NAME Global Symbol

DEC/Test Manager defines the VMS global symbol `DTM$COLLECTION_NAME` to be the current collection name before the collection prologue file executes. It is available for use in any prologue, epilogue, or template file in the collection.

For example, you can obtain a quick report of the status of a collection at the end of the run, and you can be informed when the collection is finished, by having the collection epilogue file do the following:

- Invoke DEC/Test Manager.
- Enter the `SHOW COLLECTION` command with the `/FULL` qualifier using `DTM$COLLECTION_NAME` to specify the collection name.
- Invoke the VMS Mail Utility (`MAIL`) to send you the output of this command.

See Example 6–4 for the sample collection epilogue file.

---

#### 6.3.3.2 DTM\$TEST\_NAME Local Symbol

DEC/Test Manager establishes the VMS local symbol `DTM$TEST_NAME` as the test name field of the test description. You can use `DTM$TEST_NAME` in template files, in test epilogue files, and in test prologue files.

The following example shows how you can write the file `MAIL_TEMPLATE.COM` (the template file associated with test description `SEND_MAIL_TEST`) using `DTM$TEST_NAME`. If you create a modified copy of `SEND_MAIL_TEST` (the test description that previously used template file `MAIL_TEMPLATE.COM`) and call the modified copy `REPLY_MAIL_TEST`, you can generalize `MAIL_TEMPLATE.COM` to run with both `SEND_MAIL_TEST` and `REPLY_MAIL_TEST` by using `DTM$TEST_NAME` in the template file. Example 6–7 shows the more general template file.

### Example 6–7: Using the DTM\$TEST\_NAME Local Symbol

---

```
$!      --- MAIL_TEMPLATE.COM ---
$! TEMPLATE file for MAIL message sending commands
$!
$ @'dtm$test_name'.COM
$!
$! This new template file can be used with any test whose test
$! name is the same as that of the input file.
```

---

#### 6.3.3.3 DTM\$RESULT Logical Name

DEC/Test Manager establishes the VMS logical name DTM\$RESULT as the logical equivalent to the file specification for the test result file. DEC/Test Manager defines DTM\$RESULT immediately after the test template file executes and just before the test epilogue file executes. It is deassigned after the test epilogue file executes and therefore exists only during test epilogue file execution.

DTM\$RESULT enables you to create the epilogue file to filter run-dependent information from the result file. To do this, the epilogue file runs the result file through a text editor, such as EDT.

Example 6–8 shows an epilogue file that invokes EDT to remove from the result file all lines that contain VMS run information on the amount of memory used. The epilogue file deletes all lines containing the phrase “Memory Used:”.

#### NOTE

DEC/Test Manager DECwindows result files are in DDIF format. Attempts to alter these files may corrupt them.

### Example 6–8: Using the DTM\$RESULT Logical Name

---

```
$! MEM.FIL -- Eliminate any "Memory Used:"
$!      messages from .RES files.
$ EDIT/EDT DTM$RESULT
c;32767('Memory Used:' dl) ex
EXIT
$ PURGE DTM$RESULT
```

---



---

### 6.3.3.4 DTM\$DECW\$DISPLAY Logical Name

You can define the DTM\$DECW\$DISPLAY logical name to identify the DECwindows display that DEC/Test Manager is to use for recording or playing DECwindows tests. The DTM\$DECW\$DISPLAY logical name is defined in the same way the DECW\$DISPLAY logical name in VMS is defined. See the *VMS DECwindows User's Guide* for more information about defining the DECW\$DISPLAY logical name.

For example, you can set the DEC/Test Manager record and playback workstation as the WSA1 workstation device by specifying the following command:

```
$ DEFINE DTM$DECW$DISPLAY _WSA1:
```

By default, DEC/Test Manager uses the default DECwindows display, generally determined by the DECW\$DISPLAY logical name. The /DISPLAY qualifier overrides all display determination options on the RECORD and PLAY commands.

You can use DECW\$DISPLAY to record or play DECwindows tests on another workstation.

If a command is associated with a DECwindows test, you must ensure that any applications the command may invoke connects to the DECwindows display that DEC/Test Manager will use for record or playback functions.

---

### 6.3.3.5 DTM\$DELAY\_TIMEOUT Logical Name

During interactive terminal test playback, DEC/Test Manager sends terminal input to an application as fast as the application will accept it. Under some circumstances, an application may never directly request input (with a QIO), but may check input queues periodically for the presence of input.

For interactive terminal test play back, DEC/Test Manager cannot detect when the application is ready for input, but by default, waits 7 seconds and then sends input, thus allowing the test to continue.

For DECwindows tests, test synchronization can be accomplished by editing an input file (see Chapter 9). When synchronization points are encountered in a DECwindows session file, DEC/Test Manager waits for a specified DECwindows display event to occur before continuing to send input. DEC/Test Manager waits 7 seconds, by default, but you can specify another timeout value.

You can use `DTM$DELAY_TIMEOUT` to specify your own delay timeout value. The value must be specified as a standard VMS delta time. For example, to set a timeout value of 15 seconds, enter the following logical definition:

```
$ DEFINE DTM$DELAY_TIMEOUT "0 00:00:15.0"
```

---

### 6.3.3.6 DTM\$OMIT\_PRINTABLE\_SCREEN Logical Name

When DEC/Test Manager compares interactive tests while running a collection, DEC/Test Manager can optionally create printable versions of the result (`.RES_SCREEN`) and the benchmark (`.BMK_SCREEN`) files. The files are only used for printing by the `PRINT` command with the `/RESULT` `/BENCHMARK` qualifiers during the subsequent `REVIEW` of the collection test results. The files can grow to be quite large depending on the quantity of screens compared during the collection run.

DEC/Test Manager creates printable screens files when you specify to do so based on the value of the logical name, `DTM$OMIT_PRINTABLE_SCREEN`, which can be set in the collection prologue procedure.

If the logical name is not defined or set to a value of 0, the default action is to create the printable screens. If the logical name is defined and set to a value of 1, the printable screens are not created during the comparison of the test results. Enter the following logical definition to omit creating the printable screens:

```
$ DEFINE DTM$OMIT_PRINTABLE_SCREEN 1
```

---

## 6.4 Using Filters

DEC/Test Manager enables you to filter data that varies in test results from one test run to the next. DEC/Test Manager also enables you to filter data during the recording of a test to produce a filtered benchmark file.

DEC/Test Manager filters operate by changing specified data types (like time stamps) to ASCII characters of a standard format. For example, a VMS time stamp of `13:20:23.0002` can be changed to `hh:mm:ss.xxxx`.

The following commands provide ways to specify filtering:

- `CREATE TEST_DESCRIPTION/FILTER=keyword`  
`MODIFY TEST_DESCRIPTION/FILTER=keyword`

Using the `CREATE TEST_DESCRIPTION` or `MODIFY TEST_DESCRIPTION` command with the `/FILTER` qualifier, the specified filters are to be applied only to the test being created. See Section 6.4.1. When a test is run in a collection, the filters associated with it are applied to the result file.

- **RECORD** `testname/FILTERS`

Using the `RECORD` command with the `/FILTERS` qualifier when the test is recorded causes DEC/Test Manager to apply the filters associated with the test description to the benchmark file (if a benchmark file is produced).

- **FILTER** `file-specification/qualifier`

Using the `FILTER` command, the specified filters are to be applied to the specified file, which can be any VMS file and does not necessarily need to be applied to DEC/Test Manager files. See Section 6.4.2.

Test result files are filtered when the test is run in a collection. The filtering is performed after the test epilogue file has been run.

See the `CREATE TEST_DESCRIPTION`, `FILTER`, and `MODIFY TEST_DESCRIPTION` commands in the Command Dictionary for the types of filters you can use.

DECwindows tests cannot have filters associated with them. DECwindows tests use the Mask Editor to create areas that cause DEC/Test Manager to ignore image data in screen comparisons. See Chapter 2 for information on filtering DECwindows result and benchmark files.

#### **NOTE**

Use caution with filters because the original unfiltered result file is deleted after the filtering occurs, leaving only the filtered file. Using filters on interactive tests that contain escape sequences can delete information that is essential to the test.

---

### **6.4.1 Associating and Disabling Test Filters**

Use the `CREATE TEST_DESCRIPTION` or `MODIFY TEST_DESCRIPTION` command with the `/FILTER` qualifier to associate filters with a specific test description.

When the test is executed, DEC/Test Manager filters the result file (after the epilogue file is run).

Use the `MODIFY TEST_DESCRIPTION` command with the `/NOFILTER` qualifier to disable any specified filter from a specific test. The following example shows how to remove the `DATE` filter from the `MAIL_TEST` test description:

```
DTM> MODIFY TEST_DESCRIPTION MAIL_TEST/NOFILTER=DATE
_Remark: Disabling the DATE filter
```

### NOTE

If you modify filters for a test description, you must subsequently use the `RECREATE` command to re-create any collections containing the test.

The `SHOW TEST_DESCRIPTION` with the `/FULL` or `/FILTER` qualifier list the filters associated with a specific test description.

---

## 6.4.2 Applying File Filters

To apply any or all of the filters to a file (inside or outside of a DEC/Test Manager library), use the `FILTER` command. The following command filter the time and date from the benchmark file of the `MAIL_TEST` test:

```
DTM> FILTER MAIL_TEST.BMK/TIME/DATE "Filter out time and date stamps"
```

---

## 6.5 Defining Keypad Keys

DEC/Test Manager supplies you with four default operations keypads for the terminal environment, depending on the DEC/Test Manager subsystem that you are using. Keypad definitions are not recognized by the DEC/Test Manager DECwindows interface. This section describes the DEC/Test Manager system default keypad. The other keypads are described in Chapter 5.

The DEC/Test Manager keypad is available when you invoke DEC/Test Manager. The `GOLD` key (PF1), the `HELP` key (PF2), and the `ENTER` (RETURN) keys are already defined; the rest of the keys on the keypad are undefined. You can define a key to execute up to two DEC/Test Manager commands by specifying one command to execute when you press the defined key and by specifying another command to execute when you press the `GOLD` key and then the defined key. You can create a custom keypad by defining keys to execute often-used commands or command strings that are very long.

When you create key definitions with the **DEFINE/KEY** command, these definitions are in effect only for the current DEC/Test Manager session. The next time you invoke DEC/Test Manager, only the default key definitions will be in effect. To save key definitions and to use them in every DEC/Test Manager session you initiate, include the key definitions in a DEC/Test Manager initialization file. This file is executed whenever you invoke DEC/Test Manager. For more information on initialization files, see Section 6.6.2.

If you have key definitions that you want to save but do not necessarily want to use every time you invoke DEC/Test Manager, store them in a command file.

You can define the keypad keys to execute DEC/Test Manager commands in a single keystroke by using the **DEFINE/KEY** command. The following example defines **KP5** to set the default DEC/Test Manager library:

```
DTM> DEFINE/KEY KP5 "SET LIBRARY DUA0:[USER01.LIB_A]"/TERMINATE
DTM>
```

If you subsequently press **KP5**, the following text is displayed:

```
DTM> SET LIBRARY DUA0:[USER01.LIB_A]
%DTM-S-LIBIS, DEC/Test Manager library is DUA0:[USER01.LIB_A]
DTM>
```

**GOLD** command keys are the same as regular command keys except that you must press the **GOLD** key (**PF1**) before pressing the command key. This enables you to have two commands associated with one keypad key. You can define the **GOLD** keypad keys to execute DEC/Test Manager commands in two keystrokes by using the **DEFINE/KEY** command with the **/SET\_STATE=GOLD\_DTM** qualifier. The following example defines **GOLD KP5** to set the default DEC/Test Manager library to a different library from the one in the previous example:

```
DTM> DEFINE/KEY KP5 /IF_STATE=GOLD_DTM/TERM -
_DTM> "SET LIBRARY DUA0:[USER01.LIB_B]"/TERMINATE
DTM>
```

If you subsequently press **GOLD KP5**, the following text is displayed:

```
DTM> SET LIBRARY DUA0:[USER01.LIB_B]
%DTM-S-LIBIS, DEC/Test Manager library is DUA0:[USER01.LIB_B]
DTM>
```

See the **DEFINE/KEY** command in the Command Dictionary for more information on defining keys.

---

## 6.6 Using Command Files

A DEC/Test Manager **command file** is a file containing one or more DEC/Test Manager commands. A DEC/Test Manager command file has a file type of .COM and is executed by using the @ character. The format for executing a command file is as follows:

```
DTM> @file-specification
```

When you invoke a command file, its commands execute in sequence. You can nest command files within command files. When a command file encounters a nested command file, DEC/Test Manager stops processing the original command file and begins executing the newly encountered command file. When DEC/Test Manager completes the nested command file, DEC/Test Manager resumes processing of the original command file.

### NOTE

DEC/Test Manager does not check for recursive command files. If you have a command file that invokes itself, or invokes another command file that invokes the original command file, you will create an infinite loop.

An EXIT command in a command file causes DEC/Test Manager to terminate the subsystem that is running. It does not necessarily terminate execution of the command file. For example, if the command file issues the EXIT command from the Review subsystem, control returns to the DEC/Test Manager level. If an EXIT command terminates the DEC/Test Manager session, execution of the command file terminates.

If the command file causes an error or warning to occur, execution of the command file stops and no subsequent commands are executed.

---

### 6.6.1 Creating and Invoking a Command File

You create DEC/Test Manager command files with a text editor and invoke them from the DEC/Test Manager subsystem level, from the Review subsystem level, or from within another command file. If you include only a file name with the @file-specification command, DEC/Test Manager assumes a file type of .COM.

In a DEC/Test Manager command file, if you specify a DEC/Test Manager command but omit a required parameter, DEC/Test Manager prompts you for the missing parameter.

When you invoke DEC/Test Manager from a DCL command procedure, be sure to supply all required command parameters. If you omit a required parameter for which you would be prompted if you entered the command interactively, DCL reads the next line in the command file as the missing parameter rather than as a separate command. The second command is lost.

---

## 6.6.2 Creating a DEC/Test Manager Initialization Command File

Use the DEFINE command to define a DEC/Test Manager initialization command file. DEC/Test Manager provides the VMS logical name DTM\$INIT that you define to identify a command file that you want DEC/Test Manager to execute each time you invoke DEC/Test Manager.

A typical initialization command file would contain the commands you enter every time you invoke DEC/Test Manager. For example, it could contain the command to select a DEC/Test Manager library and the commands to define keys on the DEC/Test Manager keypad. Example 6–9 establishes the DEC/Test Manager library as [USER01.DTMLIB] and defines several keypad keys on the DEC/Test Manager keypad.

### Example 6–9: Sample Initialization Command File

---

```
!Initialization file to set library and define keys
!
!Establish the library
!
SET LIBRARY DUA0:[USER01.DTMLIB]
!
!Define keypad keys
!
DEFINE/KEY KP3/IF_STATE=DTM          "SHOW COLLECTION */FULL"/TERMINATE
DEFINE/KEY PF4/IF_STATE=GOLD_DTM     "SET LIBRARY DUA0:[USER01.LIB_A]"/TERMINATE
DEFINE/KEY KP1/IF_STATE=REVIEW       "NEXT/SUCCESSFUL"/TERMINATE
DEFINE/KEY KP1/IF_STATE=GOLD_REVIEW  "BACK/SUCCESSFUL"/TERMINATE
```

---

You can suppress the initialization command file execution by invoking DEC/Test Manager with the /NOINIT qualifier, as shown in the following example:

```
$ DTM/NOINIT
```

---

## 6.7 Spawning or Attaching to Another Process

You can use the SPAWN or ATTACH commands at both the DEC/Test Manager prompt (DTM>) and the Review prompt (DTM\_REVIEW>). These commands enable you to create one or more subprocesses of your parent process, and to move between these processes.

### NOTE

The SPAWN and ATTACH commands have no corresponding action in a DECwindows environment.

The SPAWN command enables you to create (spawn) a subprocess and to attach your terminal or workstation to the subprocess. You can create a subprocess to issue DCL commands, to read an electronic mail message, or to create another DEC/Test Manager session. The ATTACH command enables you to switch to other subprocesses.

If you specify a DCL command as a parameter to the SPAWN command, the DCL command is executed and control is returned immediately to the DEC/Test Manager session. If you do not include a DCL command, the DCL prompt displays, and you can then issue DCL commands. As each command terminates, the DCL prompt reappears. You can return to the parent process by logging out of the subprocess or by issuing the ATTACH command.



# Maintaining a DEC/Test Manager Library

---

This chapter describes various methods and commands for maintaining a DEC/Test Manager library; it provides information on the following topics:

- Correcting an invalid DEC/Test Manager library
- Storing files outside a DEC/Test Manager library
- How to set up security features for a DEC/Test Manager library and its files

---

## 7.1 Correcting an Invalid DEC/Test Manager Library

If an abrupt process, termination, job termination, or a system failure occurs while a DEC/Test Manager library-altering command is executing, the library is considered invalid. DEC/Test Manager can detect errors in the library structure, its files, and its collections. DEC/Test Manager enables you to recover from these errors in structure.

To recover an invalid library, you must verify its structure by specifying the `VERIFY` command.

When you enter the `VERIFY` command to recover an invalid library, DEC/Test Manager performs an evaluation on the current DEC/Test Manager library to ensure that the library and its files have a valid structure. The `VERIFY` command also consolidates disk space.

If the library is valid, the command executes successfully. You can use the `VERIFY` command on valid libraries; it does not damage a properly created DEC/Test Manager library.

If you specify the `VERIFY` command with the `/RECOVER` qualifier, and DEC/Test Manager fails to return the library to a valid state, you must restore it from backup.

If DEC/Test Manager encounters subdirectories that are not associated with a current collection while restoring the library, you are prompted for confirmation of deletion of these subdirectories. You should not create subdirectories in any DEC/Test Manager library.

If a collection run is terminated other than with the STOP command or by pressing CTRL/C, that collection is locked and you are not able to review it. In this case, the VERIFY command with the /RECOVER qualifier unlocks the library and marks the tests that did not run.

After entering the VERIFY command with the /RECOVER qualifier, compare and then review the collection. While reviewing the collection, you can create a group containing the tests that did not run. After leaving the Review subsystem, you can execute that group of tests.

If you issue the VERIFY command with the /REPAIR qualifier, DEC/Test Manager attempts to reclaim loose blocks in the current DEC/Test Manager library and deletes illegal files found in the library.

---

## 7.2 Storing Files Outside a DEC/Test Manager Library

You must store template files, test prologue and epilogue files, and collection prologue and epilogue files outside the DEC/Test Manager library, in VMS directories, VAX DEC/Code Management System (CMS) libraries, or both. You can store benchmark files inside or outside a DEC/Test Manager library.

---

### 7.2.1 Setting Benchmark and Template Directories

Use the SET BENCHMARK\_DIRECTORY and SET TEMPLATE\_DIRECTORY commands to establish default benchmark and template directories for the current DEC/Test Manager library. DEC/Test Manager processes files faster when you use default benchmark and template directories rather than specifying a directory with each file name.

If you do not specify default benchmark and template directories, DEC/Test Manager uses your current default directory (SYS\$DISK:[ ]) for template files and the DEC/Test Manager library (DTM\$LIB) for benchmark files.

In a DECwindows environment, default benchmark and template directories are specified on the Create Library or Modify Library dialog box. The Create Library dialog box is shown in Chapter 2.

After you create or modify a test description, you can override the default directories for the current DEC/Test Manager library (if defaults exist) by using the `CREATE COLLECTION` command with the `/BENCHMARK_DIRECTORY` and `/TEMPLATE_DIRECTORY` qualifiers. The specified directories are used for the benchmark and template files for the collection being created.

To remove a default benchmark or template directory without replacing it, enter the `SET NOBENCHMARK_DIRECTORY` or the `SET NOTEMPLATE_DIRECTORY` command. These commands return the benchmark directory to `DTM$LIB` and the template directory to `SYS$DISK:[.]`.

---

## 7.2.2 Storing Files in CMS Libraries

DEC/Test Manager enables you to store your benchmark and template files in one or more VAX DEC/Code Management System (CMS) libraries. To do this, you must create the CMS libraries. Then use their directory specifications in DEC/Test Manager commands, as described in the following list:

- Create a directory and make it a CMS library. This directory cannot be a subdirectory of the DEC/Test Manager library. See the *Guide to VAX DEC/Code Management System* for information about setting up a CMS library.
- Include the directory specification for the CMS library in the appropriate DEC/Test Manager command (any command where you can specify a directory specification for a file).

**Table 7-1: DEC/Test Manager Action in CMS Libraries**

User Action	DEC/Test Manager Action in CMS
Specify benchmark and template files	DEC/Test Manager always evaluates directory specifications to determine whether the directory specifications refer to CMS libraries. If a directory name is a CMS library, DEC/Test Manager issues the appropriate CMS commands to access the files.

(continued on next page)

**Table 7-1 (Cont.): DEC/Test Manager Action in CMS Libraries**

<b>User Action</b>	<b>DEC/Test Manager Action in CMS</b>
	DEC/Test Manager translates the file name into a CMS element name and accesses the appropriate CMS element. If a CMS class is also specified with the /CLASS qualifier with the CREATE COLLECTION command, DEC/Test Manager accesses the indicated generation of the element.
Execute a collection	DEC/Test Manager fetches a copy of the template element from the CMS library and deletes the copy after using it.
Compare a collection	DEC/Test Manager compares the result file with the specified benchmark element in the CMS library. If a CMS class is specified for the benchmark element, DEC/Test Manager compares the result file with the appropriate generation of the benchmark element.
Update an existing benchmark file from the Review subsystem	DEC/Test Manager retrieves and reserves the specified benchmark element from the CMS library. DEC/Test Manager then replaces the reserved benchmark element with the result file from the current test run.  If you specify a CMS class for the benchmark file, DEC/Test Manager places the result file in the appropriate benchmark generation. If you update a benchmark generation other than the latest, a variant line D is created. If the variant line D already exists, the update fails and DEC/Test Manager unreserves the benchmark element.
Create a new benchmark file	DEC/Test Manager creates a new benchmark element in the CMS library.
Print or display a benchmark file	DEC/Test Manager fetches a copy of the benchmark element from the CMS library and deletes the benchmark file copy after printing or displaying it.
Record an interactive test whose template and benchmark files are stored in CMS libraries	DEC/Test Manager creates new elements for new benchmark and template files if the files do not already exist. If they do exist, DEC/Test Manager replaces them with new benchmark and template files.

If the directory specification portion of the file specification identifies a CMS library, DEC/Test Manager fetches a copy of the latest generation of the specified prologue or epilogue file from the CMS library. You cannot specify CMS classes for prologue and epilogue files.

---

## 7.3 Security Features

You can protect DEC/Test Manager files and libraries with two mechanisms: **user identification code (UIC)** based protection and **access control list (ACL)** protection. You protect the files and libraries; DEC/Test Manager does not define protection for library directories or library files.

UIC protection grants or denies access based on a user's UIC code. ACL protection grants or denies access based on a list of users. With ACLs you can specify access for a set of users who are not in the same UIC group.

The following procedure shows the steps VMS performs in determining access to a library directory or library file:

1. Evaluate the ACL (if present).
2. If no ACL is present, or the ACL does not prohibit access, evaluate the UIC.
3. Evaluate the privileges; if a user has GRPPRV, READALL, and SYSPRV privileges, or if a user has BYPASS privileges, VMS grants access even if the ACL and UIC protections deny access.

---

### 7.3.1 Assigning UIC Protection

Every file has a user identification code protection associated with it. UIC protection is determined by an owner UIC and a protection code. UIC protection controls access to directories and files. When you attempt to gain access to a directory or file, the system checks for existing ACLs; it then checks the UIC protection code.

See the *VMS DCL Concepts Manual* or the *VMS DCL Dictionary* for more information about UIC protection.

You can use the Digital Command Language (DCL) command SET PROTECTION to specify a particular protection setting for the library directory and for other DEC/Test Manager files. The following example sets the protection to allow system, owner, and group access to a library but deny world access to the library contained in the [PROJ.TESTING] directory:

```
$ SET PROTECTION=(S:RWE,O:RWE,G:RWE,W) [PROJ]TESTING.DIR
```

The following example uses the SET PROTECTION command to set the protection for an individual file within the library directory.

```
$ SET PROTECTION=(S:RWED,O:RWED,G:RW,W) [PROJ.TESTING]00DTM.CON
```

To use all DEC/Test Manager commands in a library, you must have the minimum access privileges (shown in Table 7–2) defined for your process:

**Table 7–2: Privileges Required for DEC/Test Manager Library Users**

<b>Object</b>	<b>Privilege</b>
Library directory	RW
Control file (00DTM.CON)	RW
History file (00DTM.HIS)	RW
All collection subdirectories (collection-name.DIR)	RWD
All collection control files (collection-name.CON) in the collection subdirectories	RWD
Generic command file (DTM\$\$TEST_RECORD.COM) used to record collections	RD
Generic command file (DTM\$\$TEST_RUN.COM) used to run collections	RD
All other files in the library directory	RD
All other files in the collection subdirectories	RD
DEC/Test Manager images (SYS\$SYSTEM:DTM.EXE, DTMSHR.EXE, and DTM\$XTRAP.EXE)	E

If you enable read-only access to a library directory or to the library control file, users cannot make changes to the contents of the library, execute tests, review tests, or perform comparisons. If you do not enable write access to the collection control files, users cannot review collections. You should enable good read and write access to these files.

When you create the library, enable read, write, and delete access to every file in the library for at least one user. The three types of access are needed to execute the VERIFY command with the /RECOVER qualifier.

#### **NOTE**

If you have SYSPRV privileges, file protection problems may occur when you issue a DEC/Test Manager command that creates files in a directory or library owned by another user because having SYSPRV privileges changes the ownership of files that are created in a directory owned by another user.

You can restrict a person's access to library files by using a UIC protection or ACL. If you do not use ACLs, all users of a particular DEC/Test Manager library must be in the same user group. If you want to define more selective protection (where various individuals in the user group have differing access), you can use ACLs for the library directory and its files and subdirectories.

The following sections summarize the procedures you can use to define the access to a DEC/Test Manager library. For more information, see the *VMS DCL Dictionary* and the *Guide to VMS System Security*.

---

## 7.3.2 Assigning ACL Protection

An ACL consists of access control entries (ACEs) that grant or deny access to a library directory file or library file to specific users. ACLs used with library directory files enable you to define access to an entire library. ACLs used with library files enable you to establish specific control over access to library contents. Generally, VMS ACLs are used in conjunction with the standard UIC-based protection as a way to fine-tune protection.

See the *VMS DCL Dictionary* for more information on these commands. See the *Guide to VMS System Security* for more information on using ACLs and ACEs.

---

### 7.3.2.1 Using ACLs on Library Directories

Directory ACLs provide three means of controlling access to a DEC/Test Manager library:

- By controlling access to the directory file itself, as shown in the following example:

```
$ SET FILE/ACL=(IDENTIFIER=DBASEGRP,ACCESS=READ+WRITE) DTMLIB.DIR
```

This ACE grants READ and WRITE access to the directory file DTMLIB.DIR to users who have the DBASEGRP identifier.

- By specifying a default UIC protection to be assigned to each new file created in the directory. To set a specific UIC protection, use the DEFAULT\_PROTECTION keyword as the first field of an ACE, as shown in the following example:

```
$ SET FILE/ACL=(DEFAULT_PROTECTION,S:RWED,O:RWED,G:RWED) DTMLIB.DIR
```

This ACE specifies that the UIC protection (S:RWED,O:RWED,G:RWED) be applied to each new file created in the directory. (It does not affect any files that may already exist in the directory.) If no other ACEs impose stricter limitations, the system, owner, and group users are granted full use of the library.

- By specifying a default identifier-based protection to be assigned to each file created in the directory. To specify a default identifier ACE, use the `OPTIONS=DEFAULT` keyword in the second field of an ACE that is applied to a directory file, as shown in the following example:

```
$ SET FILE/ACL=(IDENTIFIER=DBASEGRP,OPTIONS=DEFAULT,ACCESS=READ+WRITE+DELETE)
```

The `OPTIONS=DEFAULT` keyword directs the operating system to duplicate this ACE in the ACL of every new file that is created in the directory. This ACE grants read, write, and delete access to users who have the DBASEGRP identifier.

### 7.3.2.2 Using ACLs on Library Files

To control library access further, you can set the file protection for each file in the library.

Table 7-3 shows a list of the DEC/Test Manager commands and the protection required for each object that the command accesses. Not all the DEC/Test Manager commands are shown, because they do not all require access privileges.

**Table 7-3: Privileges Required for Individual DEC/Test Manager Commands**

<b>Command</b>	<b>Library Directory</b>	<b>Library Control File</b>	<b>Collection Subdirectories</b>	<b>Library Files</b>	<b>Collection Files</b>
COMPARE	RW	RW	RW	RW	RWD
COPY TEST_DESCRIPTION	RW	RW		RW	
CREATE COLLECTION	RW	RW	RW	RW	RWD
CREATE GROUP	RW	RW		W	
CREATE LIBRARY	RW <sup>1</sup>	RW		W	
CREATE TEST_DESCRIPTION	RW	RW		RW	

<sup>1</sup>The directory must be empty.

(continued on next page)



**Table 7-3 (Cont.): Privileges Required for Individual DEC/Test Manager Commands**

<b>Command</b>	<b>Library Directory</b>	<b>Library Control File</b>	<b>Collection Subdirectories</b>	<b>Library Files</b>	<b>Collection Files</b>
CREATE VARIABLE	RW	RW		W	
DELETE COLLECTION	RW	RW	RD	RW	RD
DELETE GROUP	RW	RW		W	
DELETE HISTORY	RW	R		RWD	
DELETE TEST_DESCRIPTION	RW	RW		RWD	
DELETE VARIABLE	RW	RW		W	
DISPLAY	RW	RW		R	
EXIT (REVIEW)	RW	RW		RW	R
INSERT GROUP	RW	RW		W	
INSERT TEST_DESCRIPTION	RW	RW		W	
MODIFY GROUP	RW	RW		W	
MODIFY TEST_DESCRIPTION	RW	RW		RWD	
MODIFY VARIABLE	RW	RW		W	
RECORD	RW	RW		W	
RECREATE	RW	RW	RWD	RWD	RWD
REMARK	RW			W	
REMOVE GROUP	RW	RW		W	
REMOVE TEST_DESCRIPTION	RW	RW		W	
REVIEW	RW	RW	R	RWD	RWD
RUN	RW	RW	RW	RW	RWD
SET BENCHMARK_DIRECTORY	RW	RW		RW	
SET EPILOGUE	RW	RW		RW	
SET LIBRARY	R	R		RW	
SET PROLOGUE	RW	RW		RW	
SET TEMPLATE_DIRECTORY	RW	RW		RW	
SHOW ALL	R	R			

(continued on next page)

**Table 7-3 (Cont.): Privileges Required for Individual DEC/Test Manager Commands**

<b>Command</b>	<b>Library Directory</b>	<b>Library Control File</b>	<b>Collection Subdirectories</b>	<b>Library Files</b>	<b>Collection Files</b>
SHOW BENCHMARK_DIRECTORY	R	R			
SHOW COLLECTION/FULL	R	R	R	R	R
SHOW EPILOGUE	R	R			
SHOW GROUP	R	R			
SHOW HISTORY	R			R	
SHOW LIBRARY	R	R			
SHOW PROLOGUE	R	R			
SHOW TEMPLATE_DIRECTORY	R	R			
SHOW TEST_DESCRIPTION	R	R			
SHOW VARIABLE	R	R			
SHOW VERSION	R	R			
STOP	RW	RW	RW	W	RW
SUBMIT	RW	RW	RW	RW	RWD
VERIFY	RW	R	R	W	R
VERIFY/RECOVER	RW	RW	RD	RW	RWD

To propagate a UIC protection, use the **DEFAULT\_PROTECTION** keyword in the first field of a directory ACE, as shown in the following example:

```
(IDENTIFIER=TESTGRP, OPTIONS=DEFAULT, S:RWED, O:RWED, G:RWED)
```

This ACE specifies that the UIC protection (S:RWED,O:RWED,G:RWED) is applied to each new file created in the directory. (It does not affect any files that may already exist in the directory.) If no ACEs impose stricter limitations, the system, owner, and group users (as defined by the UIC) are granted full use of the library.

To propagate an identifier-based protection mask, use the **OPTIONS=DEFAULT** keyword in the second field of a directory ACE. For example:

```
(IDENTIFIER=TESTGRP, OPTIONS=DEFAULT, ACCESS=READ+WRITE+DELETE)
```

The **OPTIONS=DEFAULT** keyword directs the operating system to duplicate this ACE in the ACL of every new file that is created in the library. This ACE then grants read, write, and delete access to users who have the **TESTGRP** identifier.



# Working with Terminal Session Files

---

This chapter describes session files and input files for terminal-based tests and discusses how they are used. The material in this chapter applies only to session files for interactive terminal tests. It presents information on the following topics:

- The format of session files
- The format of input files
- Creating an input file
  - from an existing session file
  - using an editor
- Recording a session file from an input file

A terminal session file contains a record of an interactive terminal session recorded for the purpose of interactively testing a program with DEC/Test Manager. A session file has the default file type `.SESSION` and contains a description of the type of terminal on which you recorded a terminal session, a record of all keystrokes you input during the terminal session, a record of all system output during the terminal session, and additional control and timing information.

An input file contains a textual representation of all or part of an interactive terminal session. You can read input files and edit them with any editor. An input file has the default file type `.INP` and contains a record of all characters you input during an interactive terminal session and **special strings**, which are textual representations of nonprinting actions (such as a backspace) and recording functions contained in session files. An input file does not contain a record of characters output for the system during the terminal session.

Input files can be used in place of manually recording a test. By editing an input file, you can change the input sent to an application during the test. You can also add wait records to control synchronization or mark additional screens for comparison.

Input files for interactive terminal tests can be used to repeat a sequence of input that may be required in multiple tests; they can also be used to perform common set-up or clean-up operations. For example:

```
DTM> RECORD testname/INPUT=SETUP.INP/APPEND
```

The previous command directs DEC/Test Manager to record the test by taking input from the file SETUP.INP and, when the file is empty, by taking input from the user.

The user can also use an input file while a terminal recording session is in progress by pressing CTRL/P-I. DEC/Test Manager prompts you for the name of the input file. When the input file has been exhausted, the user may continue recording. See Chapter 3 for more information on entering commands during a recording session.

#### NOTE

DEC/Test Manager works only with session files produced from properly generated input files and with unmodified session files it creates. If you write programs to create session files, or if you edit or otherwise modify session files, do so using input files as described in this chapter.

Session files created or modified outside of DEC/Test Manager may not be upwardly compatible with future versions of DEC/Test Manager. Session files from properly generated input files are upwardly compatible.

---

## 8.1 Terminal Session Files

A terminal session file consists of a record containing a 12-byte **terminal characteristics block** followed by a sequence of records, each beginning with a 1-character record type. Section 8.1.2.1 describes the record structure of a session file.

---

## 8.1.1 Sample Session File

The following example shows the session file from recording a terminal session where the DCL command SHOW DEFAULT is entered. Entering the RECORD command to record the session file results in the following terminal session.

```
$ DTM
DTM> CREATE TEST_DESCRIPTION
_test name: test1
_Remark: Creating a simple session file
DTM> RECORD test1
%DTM-I-DEFAULTED, benchmark file name defaulted to TEST1.BMK
%DTM-I-DEFAULTED, template file name defaulted to TEST1.SESSION
%DTM-I-BEGIN, your interactive test session is now beginning...
Type CTRL/P twice to terminate the session.
```

```
$ SHOW DEFAULT
DUA0:[USER01.DTMLIB]
$ ^P^P
```

```
^P
```

```
%DTM-I-BMK_SAVED, benchmark has been saved in file
DUA0:[USER01.DTMLIB]TEST1.BMK;1
%DTM-S-RECORDED, test TEST1 has been successfully recorded in
DUA0:[USER01.DTMLIB]TEST1.SESSION
DTM>
```

Example 8-1 shows the session file, TEST1.SESSION, which was produced by DEC/Test Manager for this interactive terminal session. Records beginning with an I indicate the input received; an O indicates the output being generated.

## Example 8-1: Sample Session File

---

```
① B`P^@<XAO>^S^A^X^D^P<X82>
② ! DTM V3.1 RECORD V3.1
O<CR>
O<KEY> (LF\TEXT)
O^@
O$
O
③ O^A
Is
Os
Ih
Oh
Io
Oo
Iw
Ow
T0 :00:01.0
I
O
Id
Od
Ie
Oe
If
Of
Ia
Oa
Iu
Ou
Il
Ol
It
Ot
I<CR>
O<CR>
O<KEY> (LF\TEXT)
O<CR>
O DUA0: [USER01.DTMLIB]
O<CR>
O<KEY> (LF\TEXT)
O<CR>
O<CR>
O<KEY> (LF\TEXT)
O^@
O$
O
③ O^A
```

---

The indicated records contain the following:

- ① The terminal characteristics block
- ② A comment record



- ③ A CTRL/A, indicating that DEC/Test Manager will compare this screen

---

## 8.1.2 Terminal Session File Structure

Because both timing and system load can affect the performance of the system and the application being tested, two or more session files recorded independently may result in completely different, yet valid, session files. This occurs even though you typed exactly the same keystrokes when recording both terminal sessions. Thus, performing a source comparison on two session files yields no useful information.

Because most VMS terminal drivers are full duplex, when dealing with session files you should consider input and output to be asynchronous events. You can be typing input while the system is simultaneously writing output. As a result, input and output may appear mixed in a session file.

For example, if you were to type input such as ABCDEF, first you would type A and a few milliseconds later the system would output A. Then you would type B and the system would output B, and so on. The corresponding session file contains a record showing that you typed A, followed by records showing that the system output A, you typed B, the system output B, and so on. This sequence is shown in the following partial session file.

```
.  
.  
.  
Ia  
Oa  
Ib  
Ob  
Ic  
Oc  
Id  
Od  
Ie  
Oe  
If  
Of  
.  
.  
.
```

Both the speed at which you type and system response time affect the session file. If you type quickly or if system response is slow, for example, the session file will be different from a session file recorded under different conditions. For the same input (ABCDEF), another session file recorded under different conditions might contain a record showing that you entered AB, followed by A from the system, C from you, then BC from the system, and so on. This sequence is shown in the following partial session file.

.  
. .  
Iab  
Oa  
Ic  
Obc  
Ide  
Ode  
If  
Of  
. .  
.

Although you typed the same input, ABCDEF, both times and the system echoed back the same letters, the two session files are different. Session files recorded under different circumstances will vary, but all session files are equally valid.

The session file will be further varied if the program outputs something to the screen other than what you type. For example, the program might output X whenever you type A or it might output nothing at all. Asynchronous events, for example, entering CTRL/T or a broadcast write that occurs while the system is echoing input, will change the session file.

The terminal driver may also arbitrarily place text in a buffer or split it into groups before generating output. Even though a program may have issued a 6-character QIO to output ABCDEF, the terminal driver may output the text as two groups, ABCD followed by EF; or as 6 separate characters; or as a single 6-character string. The terminal driver may even output the text appended to some previously entered text that has not yet been output. Each elementary operation of the terminal driver results in a separate record being written to the session file. Thus, each record in the session file describes a single terminal-driver operation.

DEC/Test Manager also writes additional timing and control information to the session file.

---

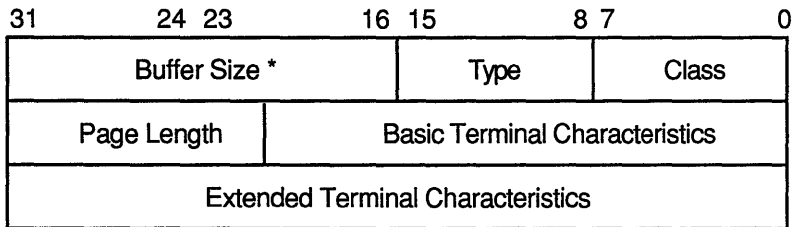
### 8.1.2.1 Record Structure of Session Files

The record structure of a session file is extremely important. You must not change it except as described in this section.

## Terminal Characteristics Block

The first record in a session file is a 12-byte (12-character) information block called a terminal characteristics block. This block of information describes the type of terminal on which the terminal session was recorded and the characteristics of that terminal. The terminal characteristics block is described in the *VMS I/O User's Reference Manual: Part 1* and is shown in Figure 8-1.

**Figure 8-1: Format of the Terminal Characteristics Block**



\* Page Width

P2 = 12

ZK-0693-GE

---

The 12 bytes in the record are stored low order to high order as three longwords. The record conforms to the structure returned by a sensemode terminal QIO with a P2 parameter of 12. The first byte has the value DC\$\_TERM.

### NOTE

You must not change the terminal characteristics block in any way. Any change to this record may invalidate the entire file. Do not add bytes to this record or delete bytes from it. This record must contain exactly 12 bytes.

## Subsequent Records

All subsequent records in the session file begin with a **record type**, which is a 1-byte indicator that describes the contents of the record. The record type is a number, usually represented by its corresponding ASCII character. For example, the decimal number 66 is referred to by the letter B. Table 8-1 shows the formats for all possible record types.

**Table 8-1: Session File Record Types**

<b>Record Type</b>	<b>Meaning</b>	<b>Description</b>
B	BEGIN_COMPARE	Restarts automatic screen compare terminated by a previous E record. When an input point is reached, DEC/Test Manager automatically marks the current screen for comparison with the corresponding screen in the benchmark file.
C	COMPARE_SCREEN	Marks the current screen for comparison, even though automatic screen comparison is turned off.
E	END_COMPARE	Terminates automatic screen compare. When an input point is reached, DEC/Test Manager automatically marks the current screen for comparison with the corresponding screen in the benchmark file.
I	INPUT	Contains characters you input, that is, characters you type at the terminal. This record usually contains only one character. These characters do not automatically echo to the screen unless the terminal is set in half-duplex mode. This input is usually echoed in a subsequent O record.
O	OUTPUT	Labels the record as containing characters output by the system and displayed on the terminal.
T	TIMING	Contains a standard VMS delta time specification. This time interval represents the clock time elapsed between the previous I (INPUT) record and the next I record.
W	WAIT	Contains a standard VMS delta time specification. It produces a pause of the specified length in the input stream when the terminal session is played back.

(continued on next page)

**Table 8-1 (Cont.): Session File Record Types**

---

<b>Record Type</b>	<b>Meaning</b>	<b>Description</b>
!	BEGIN_COMMENT	Contains a comment. This record is ignored.
0	Null or OUTPUT	This record type is supported for compatibility, though its use is not recommended. DEC/Test Manager interprets this as an O record.
1	CTRL/A or INPUT	This record type is supported for compatibility though its use is not recommended. DEC/Test Manager interprets this as an I record.

---

The T and W (TIMING and WAIT) records contain standard VMS delta time specifications. A sample time interval of 2.3 seconds would appear in a T record as follows:

```
T0 :00:02.3
```

It would appear in a W record as follows:

```
W0 :00:02.3
```

In T records, the time interval represents the elapsed clock time between the previous and next I records. DEC/Test Manager does not normally record timing intervals of less than 1 second.

When DEC/Test Manager replays a terminal session, T records cause a time delay on the input stream when DEC/Test Manager replays the session at the speed at which it was recorded.

In W records, this time interval causes a pause in the input stream of the specified duration.

---

### **8.1.2.2 Modifying Session Files Directly**

Seemingly harmless changes, such as changing the case of letters in a session file to all uppercase or all lowercase, may invalidate the entire file; this action changes the case and, therefore, the meaning of characters inside control and escape sequences.

Passing a session file through some editors (such as TECO) may change the record structure, or add or remove new line characters such as line feed or return characters. These changes invalidate the session file.

In general, splitting a record or combining two consecutive records may invalidate the entire session file. If you modify the records in a session file directly, observe the guidelines in the following sections. It is recommended that you modify session files indirectly using input files. Section 8.2 describes input files.

### **Modifying O Records**

You can combine two consecutive O records into a single O record. When you do this, drop the O character from the second record.

You can split an O record into multiple O records. When you do this, begin each new record with an O record type.

You must preserve the number of CTRL/A characters in the output stream. A single CTRL/A character represents a point where the program is requesting input and where DEC/Test Manager compares screens. Two consecutive CTRL/A characters represent an input point where automatic screen comparison is suppressed.

The terminal driver inserts a CTRL/A character into the output stream whenever the program issues an input QIO. If the program prompts for the input, the CTRL/A character appears after the prompt. Additionally, DEC/Test Manager sometimes inserts CTRL/A characters into the session file to mark input points for comparison.

### **Modifying I Records**

You can combine two consecutive I records into a single I record. When you do this, drop the I character from the second record.

You can split an I record into multiple I records. When you do this, begin each new record with an I record type.

### **Modifying B, C, and E Records**

The B, C, and E (BEGIN\_COMPARE, COMPARE\_SCREEN, and END\_COMPARE) records can occur in the session file only as the next record immediately following an O record that ends with a CTRL/A character. The one exception to this is a B, C, or E record, which can be the first noncomment record in the file immediately following the terminal characteristics block. If a B, C, or E record occurs in a session file, that file must also contain at least one O record. The number of CTRL/A characters in the file must agree with the number of CTRL/A characters output by the interactive terminal session when it is run with the PLAY command.

## Modifying T and W Records

You can modify T and W records or you can delete them to change or remove timing information. When modifying this record, you must enter the delta time in a valid format as specified in the *VMS System Services Reference Manual*. This is not necessarily in the format that DEC/Test Manager uses. Do not omit any required punctuation.

The behavior of some programs varies depending on the speed of user input. Removing or modifying timing information for such programs could adversely affect the way these programs run.

---

## 8.2 Input Files

Input files contain a textual representation of an interactive terminal session as recorded by DEC/Test Manager in a session file. Input files contain the following information:

- Input for the terminal session—the characters that were typed when you recorded the terminal session
- All nonprinting characters and recording functions represented as special strings

You can create input files in several ways. You can record an input file from an existing session file using the **EXTRACT** command, or you can create an input file using any text editor. You can also use a combination of these techniques. For example, you can edit an existing test script to reformat it as an input file.

---

### 8.2.1 Sample Input File

The following example shows the input file generated from the session file **TEST1.SESSION**, recorded in Section 8.1.1. Enter the **EXTRACT** command to generate the input file. Supply the name of the session file from which the input file is to be extracted, and designate a name for the input file.

```
DTM> EXTRACT
_session file: TEST1.SESSION
_input file: TEST1.INP

%DTM-S-EXTRACTED, input file DUA0:[USER01.DTMLIB]TEST1.INP created
DTM>
```

The following input file, TEST1.INP, was produced by the EXTRACT command:

```
SHOW DEFAULT{<CR>}
```

The input file TEST1.INP contains the text entered during the terminal session (the DCL command SHOW DEFAULT) and the special string {<CR>}, which terminates the entered text. Notice that output supplied by the system is not included in the input file. You can edit this input file and then use it to generate a new session file.

---

## 8.2.2 Special Strings

All nonprinting characters and recording functions found in session files are replaced in input files by special strings, which have the following format:

```
{special-string}
```

Special strings are textual representations for nonprinting characters and recording functions. They are enclosed within braces ({}).

---

### 8.2.2.1 Types of Special Strings Recognized by DEC/Test Manager

DEC/Test Manager recognizes the following types of special strings:

- Control and nonprinting characters—mnemonic control character name and decimal integer values for control characters available in both 7-bit and 8-bit environments
- Common names for nonprinting characters
- 8-bit control characters—mnemonic control character names and decimal integer values for control characters available only in 8-bit environments
- Key names
  - Names written on keyboard keys (with underscores substituted for spaces)
  - Names for the arrow keys
  - Names for the editing keys
  - Names for the keypad keys
  - Names for the function keys
- Names for the recording functions



When you extract an input file from a session file, DEC/Test Manager translates each nonprinting character and recording function in the session file into the appropriate special string. When you generate a session file from an input file, DEC/Test Manager retranslates the special strings. Table 8-2 lists the translations performed when extracting an input file from a session file. Tables 8-3 through 8-9 list the translations performed when recording a session file from an input file.

### Control Characters and Common Names of Nonprinting Characters

DEC/Test Manager recognizes the following formats for control characters and nonprinting characters:

- Special strings for control characters of the forms {CTRL/x} and {^x}.  
For example, DEC/Test Manager interprets both {CTRL/A} and {^A} as the same control character.
- Special strings for all mnemonic control character names listed in the ASCII 7-bit and 8-bit character tables surrounded by angle brackets (<>).  
For example, DEC/Test Manager interprets {<SOH>} and {<IND>}.
- Special strings for all integer decimal values for control characters listed in the ASCII 7-bit and 8-bit character tables.  
For example, DEC/Test Manager interprets {27} as ESC.  
DEC/Test Manager ignores leading zeros on integer decimal values.
- Special strings for common names for nonprinting characters:

{<BACK\_SPACE>}

{<DELETE>}

{<ENTER>}

{<ESC>}

{<ESCAPE>}

{<FORM\_FEED>}

{<LINE\_FEED>}

{<PAGE>}

{<RETURN>}

{SPACE}, {<SP>}, {32}, and regular spaces { } are all recognized as the space character

{<KEY> (TAB \ TEXT)}

Tables 8-3 and 8-4 list the special string translations that DEC/Test Manager performs for control characters. Table 8-3 also lists the special string translations for common names for nonprinting characters. Consult a terminal manual for tables on ASCII 7-bit and 8-bit codes.

## Key Names

DEC/Test Manager recognizes special strings for the following:

- Names written on the keyboard keys with underscores ( `_` ) substituted for spaces. For example, DEC/Test Manager recognizes `{RETURN}`, `{LINE_FEED}`, and `{PF1}`.
- Function keys by name. For example, DEC/Test Manager recognizes both `{F12}` and `{BS}` (see Table 8–5).
- Editing keys by name. For example, DEC/Test Manager recognizes `{REMOVE}`, `{NEXT_SCREEN}`, and `{NEXT}` (see Table 8–6).
- Keypad keys by name. For example, DEC/Test Manager recognizes `{KP3}`, `{MINUS}`, and `{ENTER}` (see Table 8–7).
- Arrow keys by name. For example, DEC/Test Manager recognizes both `{UP_ARROW}` and `{UP}` (see Table 8–8).

### NOTE

Tables 8–5 through 8–8, located at the end of this chapter, list the special string translations that DEC/Test Manager performs for key names.

## Recording Functions

DEC/Test Manager recognizes the following special strings for the recording functions:

```
{BEGIN_COMMENT} and {END_COMMENT}
{BEGIN_COMPARE} and {END_COMPARE}
{COMPARE_SCREEN}
{DELAY}
{WAIT}
```

Use these special strings in input files to avoid making the input file dependent on a particular termination character. For example, if you enter `{CTRL/P}C` in an input file to mark a screen for comparison, the input file and all session files generated from it must be used in conjunction with the termination character `CTRL/P`. But if you enter `{COMPARE_SCREEN}` to mark a screen for comparison, the input file and all session files generated from it can be used in conjunction with any termination character.

Table 8–9 lists the special string translations that DEC/Test Manager performs for the recording functions.

---

### 8.2.2.2 Using Special Strings in Input Files

DEC/Test Manager's interpretation of special strings in input files is not case sensitive. You can use uppercase or lowercase characters, or a combination of the two when you enter a special string in an input file.

When processing input files, DEC/Test Manager does not interpret the end of a record as the end of input. Therefore, you must be careful to enter a special string corresponding to a carriage return, (for example, {<CR>}) at the end of any input normally terminated with a carriage return.

To include text enclosed within braces ({} ) in the input file, enter double braces around the text, (for example, {{ text }}). When processing the input file, DEC/Test Manager translates the double braces to single braces; DEC/Test Manager does not interpret the text contained within the braces as a special string.

Use the special string equivalents for the recording functions when writing an input file. This avoids building a dependency on a particular termination character into an input file.

When you enter a comment in an input file, enclose the comment between the {BEGIN\_COMMENT} and {END\_COMMENT} special strings. Enter the {BEGIN\_COMMENT} and {END\_COMMENT} special strings on separate lines because DEC/Test Manager ignores the remainder of the line following these special strings. You must begin each line of comment text with the comment character (!).

Do not nest input files. DEC/Test Manager ignores INSERT recording functions (CTRL/P I) when they occur in input files.

---

## 8.3 Creating Input Files

You can create an input file using a text editor or you can use the EXTRACT command to create an input file from a session file. When the input file is complete and correctly formatted, you can then use it to record a session file.

### NOTE

When you create a terminal input file, you should use the same type of display device on which the session file was created. If you do not, you may cause the input file and session to have different characteristics.

When you extract an input file from a session file, you may not be able to re-create the session file except by using a display device of the same type as the display device used to record the original recording session, especially if the recording display device handles 8-bit characters and the extracting display device handles 7-bit characters.

---

### 8.3.1 Extracting an Input File from a Session File

The **EXTRACT** command extracts an input file from an existing session file without altering the session file. The format for the **EXTRACT** command is as follows:

```
DTM> EXTRACT session-file-specification [input-file-specification]/INTERACTIVE
```

The session file specification is the file specification of the session file from which DEC/Test Manager is to extract an input file. If you specify a file name for the session file without specifying a file type, the file type defaults to **.SESSION**.

The input file specification is the file specification for the input file being created. If you do not specify an input file specification, the file specification defaults to **session-file-name.INP**. If you specify an input file name without specifying a file type, the file type defaults to **.INP**.

You can store both files in VAX DEC/Code Management System (CMS) libraries. The **EXTRACT** command can fetch the session file from a CMS library and place the input file in the CMS library.

If you subsequently record a session file from an input file stored in a CMS library, you must first fetch the input file from the CMS library by issuing the appropriate CMS commands.

The **EXTRACT** command takes the **/[NO]LOG** and **/TERMINATION\_CHARACTER** qualifiers. The **/TERMINATION\_CHARACTER** qualifier specifies the termination character that DEC/Test Manager is to use when translating the recording functions in the session file to special strings in the input file. If the interactive terminal session you are recording does not use the default termination character (**CTRL/P**), you need not specify a different termination character. If the interactive terminal session you are recording uses **CTRL/P** for its own purposes, you must specify a different termination character.

When DEC/Test Manager extracts the input file from the session file, the following occurs:

- All input in the session file is written to the input file.

- All nonprinting characters and recording functions are translated to special strings—text delimited by braces ( { } ).
- Any braces appearing in the text of the session file are doubled in the input file.

The following example extracts the input file TEST1\_A.INP from the session file TEST1.SESSION:

```
DTM> EXTRACT TEST1.SESSION TEST1_A.INP

%DTM-S-EXTRACTED, input file DUA0:[USER01.DTMLIB]TEST1_A.INP created
DTM>
```

---

### 8.3.2 Creating an Input File with a Text Editor

You can create an input file using any text editor. See Section 8.2.2.2 for information on using special strings in input files.

The following example shows a sample input file created with a text editor.

```
{BEGIN_COMMENT}
! This is a sample input file. When used, it calls up EDT
! to create a file, enters some text into the buffer, and
! moves around that text, finally quitting without saving
! the file.
{END_COMMENT}
edit/edt sample.tmp{<CR>}
change{<CR>}
{BEGIN_COMMENT}
! Enter the text into the buffer.
{END_COMMENT}
This is the first line of the file.{<CR>}
This is the
second line of the file.{<CR>}
{UP_ARROW}{UP}{KP2}
{SPACE}This is more of the first line.
{CTRL/Z}
quit{<CR>}
```

---

## 8.4 Recording a Session File from an Input File

You can record a session file from an input file in two ways:

- By using the /INPUT qualifier on a RECORD command
- By entering the INSERT recording function (CTRL/P I) while recording an interactive terminal session

---

## 8.4.1 Using the /INPUT Qualifier

The RECORD command with the /INPUT qualifier specifies that DEC/Test Manager record a new session file by initiating an interactive terminal session and taking input from the specified input file. If you do not also specify the /APPEND qualifier, DEC/Test Manager terminates the interactive terminal session when the input file is exhausted. If you include both the /INPUT and /APPEND qualifiers, DEC/Test Manager leaves the terminal in record mode when the input file is exhausted. You can then continue the terminal session interactively and terminate it by entering the termination character (CTRL/P) twice.

The following example initiates an interactive terminal session to create session file TEST2.SESSION, takes all input from input file TEST2.INP, and terminates the terminal session when TEST2.INP is exhausted.

```
DTM> RECORD/INPUT=TEST2.INP
  Remark: Recording TEST2.SESSION from TEST2.INP
%DTM-I-DEFAULTED, benchmark file name defaulted to TEST2.BMK
%DTM-I-DEFAULTED, template file name defaulted to TEST2.SESSION
%DTM-I-BEGIN, your interactive test session is now beginning...
```

```
$ show default
  DUA0:[USER01.DTMLIB]
```

```
^P
```

```
%DTM-I-BMK_SAVED, benchmark has been saved in file
  DUA0:[USER01.DTMLIB]TEST2.BMK;1
%DTM-S-RECORDED, test TEST2 has been successfully recorded in
  DUA0:[USER01.DTMLIB]TEST2.SESSION
DTM>
```

The following example initiates an interactive terminal session to create the session file SAMPLE\_TEST.SESSION, takes input from the input file SAMPLE\_TEST.INP until the file is exhausted, and leaves the terminal in record mode:

```

DTM> RECORD/INPUT=SAMPLE_TEST.INP
_Remark: Recording a sample session file
%DTM-I-DEFAULTED, benchmark file name defaulted to SAMPLE_TEST.BMK
%DTM-I-DEFAULTED, template file name defaulted to SAMPLE_TEST.SESSION
%DTM-I-BEGIN, your interactive test session is now beginning...
Type CTRL/P twice to terminate the session.

$ show default
  DUA0:[USER01.DTMLIB]
$ show time

$ ^P^P

^P

%DTM-I-BMK SAVED, benchmark has been saved in file
  DUA0:[USER01.DTMLIB]SAMPLE_TEST.BMK;1
%DTM-S-RECORDED, test SAMPLE_TEST has been successfully recorded in
  DUA0:[USER01.DTMLIB]SAMPLE_TEST.SESSION
DTM>

```

---

## 8.4.2 Using the INSERT Recording Function

When you enter the INSERT recording function CTRL/P-I during an active recording session, it specifies that DEC/Test Manager is to take input from the specified input file. When you enter the INSERT recording function, DEC/Test Manager prompts you for the file specification for a single input file. DEC/Test Manager then takes input from the specified input file and returns control to the terminal when the input file is exhausted.

Example 8-2 shows how to insert an input file during a recording session.

During a terminal session, you can read input from multiple input files sequentially. The input files cannot be nested. DEC/Test Manager ignores any INSERT recording functions in an input file.

---

## 8.4.3 Terminal Characteristics

The process of creating an input file is not terminal specific. When you extract an input file from a session file, all control codes and other nonprinting characters are translated to special strings regardless of whether they have meaning to the terminal you are using to perform the translation.

## Example 8-2: Inserting an Input File into a Recording Session

---

```
DTM> RECORD/INPUT=SAMPTEST.INP
_test name: SAMPTEST
_Remark: Recording SAMPTEST.SESSION from SAMPTEST.INP
%DTM-I-DEFAULTED, benchmark file name defaulted to SAMPTEST.BMK
%DTM-I-DEFAULTED, template file name defaulted to SAMPTEST.SESSION
%DTM-I-BEGIN, your interactive test session is now beginning...
Type CTRL/P twice to terminate the session.

.
.
.

^PI

_Input file: SAMPTEST2.INP

.
.
.

^P^P

^P

%DTM-I-BMK SAVED, benchmark has been saved in file
DUA0:[USER01.DTMLIB]SAMPTEST.BMK;1
%DTM-S-RECORDED, test SAMPTEST has been successfully recorded in
DUA0:[USER01.DTMLIB]SAMPTEST.SESSION
DTM>
```

---

The process of creating a session file from an input file is terminal specific. When you record a session file, DEC/Test Manager translates all special strings back to control codes and other nonprinting characters based on the terminal characteristics for the recording terminal. If DEC/Test Manager encounters a special string that it cannot translate for the recording terminal, the braces are stripped off the special string and the characters representing the untranslated special string are printed in the session file. They are also displayed on the terminal screen along with an error message

When you record a session file, the terminal characteristics for the recording terminal become the first record of that session file. That session file is guaranteed to run on terminals of the same type as the recording terminal. The session file may or may not run on other terminal types.



## NOTE

You may encounter problems rerecording a session file on a VT100-series terminal if the session file was originally recorded on a VT200-series terminal. Problems will occur if the original session file contains control codes that are restricted to use in an 8-bit compatible environment. Table 8–4 lists these codes. DEC/Test Manager cannot translate these 8-bit control codes for use in a 7-bit environment.

If you record an interactive terminal session on a VT100-series terminal and extract an input file from this session file, you will be able to successfully record the edited terminal session again on either a VT100- or VT200-series terminal.

---

### 8.4.4 Type-Ahead

Anything you type on a terminal while input is being taken from an input file will have no immediate effect on the terminal session. All or part of what you type may be stored as type-ahead and may appear when the input file is exhausted and control is returned to the terminal.

---

## 8.5 Translation Tables

Table 8–2 describes translation of nonprinting characters and control codes when an input file is extracted from a session file.

**Table 8–2: Translation of Nonprinting Characters and Control Codes When Extracting an Input File from a Session File**

Code in Session File			Translated Special String in Input File	
Mnemonic	Recording Function	8-bit Control String	Escape Sequence	Special String
—	—	<CSI> A and <SS3>A	<ESC> A and <ESC>[ A and <ESC>O A	{UP_ARROW}

(continued on next page)

**Table 8–2 (Cont.): Translation of Nonprinting Characters and Control Codes When Extracting an Input File from a Session File**

Code in Session File			Translated Special String in Input File	
Mnemonic	Recording Function	8-bit Control String	Escape Sequence	Special String
—	—	<CSI> B and <SS3> B	<ESC> B and <ESC>[ B and <ESC>O B	{DOWN_ARROW}
—	—	<CSI> C and <SS3> C	<ESC> C and <ESC>[ C and <ESC>O C	{LEFT_ARROW}
—	—	<CSI> D and <SS3> D	<ESC> D and <ESC>[ D and <ESC>O D	{RIGHT_ARROW}
—	—	<SS3> P	<ESC> P and <ESC>O P	{PF1}
—	—	<SS3> Q	<ESC> Q and <ESC>O Q	{PF2}
—	—	<SS3> R	<ESC> R and <ESC>O R	{PF3}
—	—	<SS3> S	<ESC> S and <ESC>O S	{PF4}
—	—	<SS3> l	<ESC>? l and <ESC>O l	{COMMA}
—	—	<SS3> m	<ESC>? m and <ESC>O m	{MINUS}
—	—	<SS3> n	<ESC>? n and <ESC>O n	{PERIOD}
—	—	<SS3> M	<ESC>? M and <ESC>O M	{ENTER}
—	—	<SS3> p	<ESC>? p and <ESC>O p	{KP0}
—	—	<SS3> q	<ESC>? q and <ESC>O q	{KP1}

(continued on next page)

**Table 8–2 (Cont.): Translation of Nonprinting Characters and Control Codes When Extracting an Input File from a Session File**

Code in Session File				Translated Special String in Input File
Mnemonic	Recording Function	8-bit Control String	Escape Sequence	Special String
—	—	<SS3> r	<ESC>? r and <ESC>O r	{KP2}
—	—	<SS3> s	<ESC>? s and <ESC>O s	{KP3}
—	—	<SS3> t	<ESC>? t and <ESC>O t	{KP4}
—	—	<SS3> u	<ESC>? u and <ESC>O u	{KP5}
—	—	<SS3> v	<ESC>? v and <ESC>O v	{KP6}
—	—	<SS3> w	<ESC>? w and <ESC>O w	{KP7}
—	—	<SS3> x	<ESC>? x and <ESC>O x	{KP8}
—	—	<SS3> y	<ESC>? y and <ESC>O y	{KP9}
—	—	<CSI> 17~	<ESC>[ 17~	{F6}
—	—	<CSI> 18~	<ESC>[ 18~	{F7}
—	—	<CSI> 19~	<ESC>[ 19~	{F8}
—	—	<CSI> 21~	<ESC>[ 21~	{F10}
—	—	<CSI> 23~	<ESC>[ 23~	{F11}
—	—	<CSI> 24~	<ESC>[ 24~	{F12}
—	—	<CSI> 25~	<ESC>[ 25~	{F13}
—	—	<CSI> 26~	<ESC>[ 26~	{F14}
—	—	<CSI> 28~	<ESC>[ 28~	{F15}
—	—	<CSI> 29~	<ESC>[ 29~	{F16}
—	—	<CSI> 31~	<ESC>[ 31~	{F17}

(continued on next page)

**Table 8-2 (Cont.): Translation of Nonprinting Characters and Control Codes When Extracting an Input File from a Session File**

Code in Session File				Translated Special String in Input File
Mnemonic	Recording Function	8-bit Control String	Escape Sequence	Special String
—	—	<CSI> 32~	<ESC>[ 32~	{F18}
—	—	<CSI> 33~	<ESC>[ 33~	{F19}
—	—	<CSI> 34~	<ESC>[ 34~	{F20}
—	—	<CSI> 1~	<ESC>[ 1~	{FIND}
—	—	<CSI> 2~	<ESC>[ 2~	{INSERT_HERE}
—	—	<CSI> 3~	<ESC>[ 3~	{REMOVE}
—	—	<CSI> 4~	<ESC>[ 4~	{SELECT}
—	—	<CSI> 5~	<ESC>[ 5~	{PREV_SCREEN}
—	—	<CSI> 6~	<ESC>[ 6~	{NEXT_SCREEN}
<NUL>	—	—	—	{CTRL/@}
<SOH>	—	—	—	{CTRL/A}
<STX>	—	—	—	{CTRL/B}
<ETX>	—	—	—	{CTRL/C}
<EOT>	—	—	—	{CTRL/D}
<ENQ>	—	—	—	{CTRL/E}
<ACK>	—	—	—	{CTRL/F}
<BEL>	—	—	—	{CTRL/G}
<BS>	—	—	—	{<BS>}
<HT>	—	—	—	{<TAB>}
<LF>	—	—	—	{<LF>}
<VT>	—	—	—	{CTRL/K}
<FF>	—	—	—	{<FF>}
<CR>	—	—	—	{<CR>}
<SO>	—	—	—	{CTRL/N}
<SI>	—	—	—	{CTRL/O}

(continued on next page)

**Table 8–2 (Cont.): Translation of Nonprinting Characters and Control Codes When Extracting an Input File from a Session File**

Code in Session File				Translated Special String in Input File
Mnemonic	Recording Function	8-bit Control String	Escape Sequence	Special String
<DLE>	—	—	—	{CTRL/P}
<DC1>	—	—	—	{CTRL/Q}
<DC2>	—	—	—	{CTRL/R}
<DC3>	—	—	—	{CTRL/S}
<DC4>	—	—	—	{CTRL/T}
<NAK>	—	—	—	{CTRL/U}
<SYN>	—	—	—	{CTRL/V}
<ETB>	—	—	—	{CTRL/W}
<CAN>	—	—	—	{CTRL/X}
<EM>	—	—	—	{CTRL/Y}
<SUB>	—	—	—	{CTRL/Z}
<ESC>	—	—	—	{<ESC>}
<FS>	—	—	—	{CTRL/\}
<GS>	—	—	—	{CTRL/]}]
<RS>	—	—	—	{CTRL/^}
<US>	—	—	—	{CTRL/_}
<DEL>	—	—	—	{<DEL>}
<IND>	—	—	—	{<IND>}
<NEL>	—	—	—	{<NEL>}
<SSA>	—	—	—	{<SSA>}
<ESA>	—	—	—	{<ESA>}
<HTS>	—	—	—	{<HTS>}
<HTJ>	—	—	—	{<HTJ>}
<VTS>	—	—	—	{<VTS>}
<PLD>	—	—	—	{<PLD>}

(continued on next page)

**Table 8–2 (Cont.): Translation of Nonprinting Characters and Control Codes When Extracting an Input File from a Session File**

Code in Session File				Translated Special String in Input File
Mnemonic	Recording Function	8-bit Control String	Escape Sequence	Special String
<PLU>	—	—	—	{<PLU>}
<RI>	—	—	—	{<RI>}
<SS2>	—	—	—	{<SS2>}
<SS3>	—	—	—	{<SS3>}
<DCS>	—	—	—	{<DCS>}
<PU1>	—	—	—	{<PU1>}
<PU2>	—	—	—	{<PU2>}
<STS>	—	—	—	{<STS>}
<CCH>	—	—	—	{<CCH>}
<MW>	—	—	—	{<MW>}
<SPA>	—	—	—	{<SPA>}
<EPA>	—	—	—	{<EPA>}
<CSI>	—	—	—	{<CSI>}
<ST>	—	—	—	{<ST>}
<OSC>	—	—	—	{<OSC>}
<PM>	—	—	—	{<PM>}
<APC>	—	—	—	{<APC>}
—	!++	—	—	{BEGIN_COMMENT}
—	!--	—	—	{END_COMMENT}
—	CTRL/P <sup>1</sup> B	—	—	{BEGIN_COMPARE}
—	CTRL/P <sup>1</sup> C	—	—	{COMPARE_SCREEN}
—	CTRL/P <sup>1</sup> E	—	—	{END_COMPARE}
—	CTRL/P <sup>1</sup> W	—	—	{WAIT}

<sup>1</sup>CTRL/P is used here only as an example. The actual value will be the termination character specified when the session file was recorded.

Table 8–3 describes special string translations for control and nonprinting characters when a session file is recorded from an input file. Where applicable, special strings for common names of nonprinting characters are

listed with the mnemonic for the control character. For example, {BACK\_SPACE} is listed with {CTRL/H}.

**Table 8-3: Translation of Special Strings Representing Control and Nonprinting Characters When Recording a Session File from an Input File**

Special Strings in Input File			Translation in Session File	
Control Character Mnemonics			Decimal Value	
{CTRL/@}	{^@}	{<NUL>}	{0}	NUL
{CTRL/A}	{^A}	{<SOH>}	{1}	SOH
{CTRL/B}	{^B}	{<STX>}	{2}	STX
{CTRL/C}	{^C}	{<ETX>}	{3}	ETX
{CTRL/D}	{^D}	{<EOT>}	{4}	EOT
{CTRL/E}	{^E}	{<ENQ>}	{5}	ENQ
{CTRL/F}	{^F}	{<ACK>}	{6}	ACK
{CTRL/G}	{^G}	{<BEL>}	{7}	BEL
{CTRL/H}	{^H}	{<BS>}	{8}	BS
and {BACK_SPACE}				
{CTRL/I}	{^I}	{<HT>}	{9}	HT
and {TAB}				
{CTRL/J}	{^J}	{<LF>}	{10}	LF
and {LINE_FEED}				
{CTRL/K}	{^K}	{<VT>}	{11}	VT
{CTRL/L}	{^L}	{<FF>}	{12}	FF
and {PAGE}				
and {FORM_FEED}				
{CTRL/M}	{^M}	{<CR>}	{13}	CR
and {RETURN}				
{CTRL/N}	{^N}	{<SO>}	{14}	SO
{CTRL/O}	{^O}	{<SI>}	{15}	SI

(continued on next page)

**Table 8–3 (Cont.): Translation of Special Strings Representing Control and Nonprinting Characters When Recording a Session File from an Input File**

Special Strings in Input File				Translation in Session File
Control Character Mnemonics			Decimal Value	
{CTRL/P}	{^P}	{<DLE>}	{16}	DLE
{CTRL/Q}	{^Q}	{<DC1>}	{17}	DC1
{CTRL/R}	{^R}	{<DC2>}	{18}	DC2
{CTRL/S}	{^S}	{<DC3>}	{19}	DC3
{CTRL/T}	{^T}	{<DC4>}	{20}	DC4
{CTRL/U}	{^U}	{<NAK>}	{21}	NAK
{CTRL/V}	{^V}	{<SYN>}	{22}	SYN
{CTRL/W}	{^W}	{<ETB>}	{23}	ETB
{CTRL/X}	{^X}	{<CAN>}	{24}	CAN
{CTRL/Y}	{^Y}	{<EM>}	{25}	EM
{CTRL/Z}	{^Z}	{<SUB>}	{26}	SUB
{CTRL/[] {ESCAPE} and {ESC}	{^[] {^_}	{<ESC>}	{27}	ESC
{CTRL/\}	{^\\}	{<FS>}	{28}	FS
{CTRL/}]	{^}]	{<GS>}	{29}	GS
{CTRL/~}	{^~}	{<RS>}	{30}	RS
{CTRL/?} and {CTRL/_}	{^?} and {^_}	{<US>}	{31}	US
{DELETE}	—	{<DEL>}	{128}	DEL
{SPACE}	—	—	—	(a space character)

Table 8–4 describes special string translations for 8-bit control characters when a session file is recorded from an input file. These control characters are available only in 8-bit environments.



**Table 8-4: Translation of Special Strings Representing 8-Bit Control Characters When Recording a Session File from an Input File**

Special String in Input File		Translation in Session File
Mnemonic	Decimal Value	
{<IND>}	{132}	IND
{<NEL>}	{133}	NEL
{<SSA>}	{134}	SSA
{<ESA>}	{135}	ESA
{<HTS>}	{136}	HTS
{<HTJ>}	{137}	HTJ
{<VTS>}	{138}	VTS
{<PLD>}	{139}	PLD
{<PLU>}	{140}	PLU
{<RI>}	{141}	RI
{<SS2>}	{142}	SS2
{<SS3>}	{143}	SS3
{<DCS>}	{144}	DCS
{<PU1>}	{145}	PU1
{<PU2>}	{146}	PU2
{<STS>}	{147}	STS
{<CCH>}	{148}	CCH
{<MW>}	{149}	MW
{<SPA>}	{150}	SPA
{<EPA>}	{151}	EPA
{<CSI>}	{155}	CSI
{<ST>}	{156}	ST
{<OSC>}	{157}	OSC
{<PM>}	{158}	PM
{<APC>}	{159}	APC

Table 8-5 describes special string translations for codes generated by the function keys when a session file is recorded from an input file.

**Table 8-5: Translation of Special Strings Representing the Function Key Codes When Recording a Session File from an Input File**

Special String in Input File	Translation in Session File	
	VT200 Mode	VT100 and VT52 Mode
{F6}	CSI 17~ and ESC [ <sup>1</sup> 17~	—
{F7}	CSI 18~ and ESC [ <sup>1</sup> 18~	—
{F8}	CSI 19~ and ESC [ <sup>1</sup> 19~	—
{F9}	CSI 20~ and ESC [ <sup>1</sup> 20~	—
{F10}	CSI 21~ and ESC [ <sup>1</sup> 21~	—
{F11} and {ESC}	CSI 23~ and ESC [ <sup>1</sup> 23~	ESC
{F12} and {BS}	CSI 24~ and ESC [ <sup>1</sup> 24~	BS
{F13} and {LF}	CSI 25~ and ESC [ <sup>1</sup> 25~	LF
{F14}	CSI 26~ and ESC [ <sup>1</sup> 26~	—
{F15} and {HELP}	CSI 28~ and ESC [ <sup>1</sup> 28~	—
{F16} and {DO}	CSI 29~ and ESC [ <sup>1</sup> 29~	—
{F17}	CSI 31~ and ESC [ <sup>1</sup> 31~	—
{F18}	CSI 32~ and ESC [ <sup>1</sup> 32~	—
{F19}	CSI 33~ and ESC [ <sup>1</sup> 33~	—
{F20}	CSI 34~ and ESC [ <sup>1</sup> 34~	—

<sup>1</sup>ESC [ is the 7-bit code extension equivalent for the 8-bit control string CSI.

Table 8-6 describes special string translations for codes associated with editing keys when a session file is recorded from an input file.

**Table 8–6: Translation of Special Strings Representing the Editing Key Codes When Recording a Session File from an Input File**

Special String in Input File	Translation in Session File	
	VT200 Mode	VT100 and VT52 Modes
{FIND}	CSI 1~ and ESC [ <sup>1</sup> 1~	—
{INSERT_HERE} and {INSERT}	CSI 2~ and ESC [ <sup>1</sup> 2~	—
{REMOVE}	CSI 3~ and ESC [ <sup>1</sup> 3~	—
{SELECT}	CSI 4~ and ESC [ <sup>1</sup> 4~	—
{PREV_SCREEN} and {PREV}	CSI 5~ and ESC [ <sup>1</sup> 5~	—
{NEXT_SCREEN} and {NEXT}	CSI 6~ and ESC [ <sup>1</sup> 6~	—

<sup>1</sup>ESC [ is the 7-bit code extension equivalent for the 8-bit control string CSI.

Table 8–7 describes special string translations for codes associated with the keypad keys when a session file is recorded from an input file.

**Table 8–7: Translation of Special Strings Representing the Keypad Key Codes When Recording a Session File from an Input File**

Special String in Input File	Translation in Session File			
	ANSI Mode <sup>1</sup>		VT52 Mode <sup>1</sup>	
	Numeric Keypad Mode	Application Keypad Mode	Numeric Keypad Mode	Application Keypad Mode
{KP0}	0	SS3 p and ESC O <sup>2</sup> p	0	ESC ? p
{KP1}	1	SS3 q and ESC O <sup>2</sup> q	1	ESC ? q

<sup>1</sup>ANSI mode applies to VT200 and VT100 modes. VT52 mode is an ANSI-incompatible mode.

<sup>2</sup>ESC O is the 7-bit code extension equivalent for the 8-bit control string SS3.

(continued on next page)

**Table 8-7 (Cont.): Translation of Special Strings Representing the Keypad Key Codes When Recording a Session File from an Input File**

Special String in Input File	Translation in Session File			
	ANSI Mode <sup>1</sup>		VT52 Mode <sup>1</sup>	
	Numeric Keypad Mode	Application Keypad Mode	Numeric Keypad Mode	Application Keypad Mode
{KP2}	2	SS3 r and ESC O <sup>2</sup> r	2	ESC ? r
{KP3}	3	SS3 s and ESC O <sup>2</sup> s	3	ESC ? s
{KP4}	4	SS3 t and ESC O <sup>2</sup> t	4	ESC ? t
{KP5}	5	SS3 u and ESC O <sup>2</sup> u	5	ESC ? u
{KP6}	6	SS3 v and ESC O <sup>2</sup> v	6	ESC ? v
{KP7}	7	SS3 w and ESC O <sup>2</sup> w	7	ESC ? w
{KP8}	8	SS3 x and ESC O <sup>2</sup> x	8	ESC ? x
{KP9}	9	SS3 y and ESC O <sup>2</sup> y	9	ESC ? y
{COMMA}	, (comma)	SS3 l and ESC O <sup>2</sup> l	, (comma)	ESC ? l
{MINUS}	- (minus)	SS3 m and ESC O <sup>2</sup> m	- (minus)	ESC ? m
{PERIOD}	. (period)	SS3 n and ESC O <sup>2</sup> n	. (period)	ESC ? n
{ENTER}	CR	SS3 M and ESC O <sup>2</sup> M	CR	ESC ? M
{PF1}	SS3 P and ESC O <sup>2</sup> P	SS3 P and ESC O <sup>2</sup> P	ESC P	ESC P

<sup>1</sup>ANSI mode applies to VT200 and VT100 modes. VT52 mode is an ANSI-incompatible mode.

<sup>2</sup>ESC O is the 7-bit code extension equivalent for the 8-bit control string SS3.

(continued on next page)

**Table 8–7 (Cont.): Translation of Special Strings Representing the Keypad Key Codes When Recording a Session File from an Input File**

Special String in Input File	Translation in Session File			
	ANSI Mode <sup>1</sup>		VT52 Mode <sup>1</sup>	
	Numeric Keypad Mode	Application Keypad Mode	Numeric Keypad Mode	Application Keypad Mode
{PF2}	SS3 Q and ESC O <sup>2</sup> Q	SS3 Q and ESC O <sup>2</sup> Q	ESC Q	ESC Q
{PF3}	SS3 R and ESC O <sup>2</sup> R	SS3 R and ESC O <sup>2</sup> R	ESC R	ESC R
{PF4}	SS3 S and ESC O <sup>2</sup> S	SS3 S and ESC O S <sup>2</sup>	ESC S	ESC S

<sup>1</sup>ANSI mode applies to VT200 and VT100 modes. VT52 mode is an ANSI-incompatible mode.

<sup>2</sup>ESC O is the 7-bit code extension equivalent for the 8-bit control string SS3.

Table 8–8 describes special string translations for arrow key codes when a session file is recorded from an input file.

**Table 8–8: Translation of Special Strings Representing the Arrow Key Codes When Recording a Session File from an Input File**

Special String in Input File	Translation in Session File			
	ANSI Mode <sup>1</sup>		VT52 Mode <sup>1</sup>	
	Cursor Key Mode Reset Normal	Cursor Key Mode Set Application	Cursor Key Mode Reset Normal	Cursor Key Mode Set Application
{UP_ARROW} and {UP}	CSI A and ESC [ <sup>3</sup> A	SS3 A and ESC O <sup>2</sup> A	ESC A	ESC A

<sup>1</sup>ANSI mode applies to VT200 and VT100 modes. VT52 mode is an ANSI-incompatible mode.

<sup>2</sup>ESC O is the 7-bit code extension equivalent for the 8-bit control string SS3.

<sup>3</sup>ESC [ is the 7-bit code extension equivalent for the 8-bit control string CSI.

(continued on next page)

**Table 8–8 (Cont.): Translation of Special Strings Representing the Arrow Key Codes When Recording a Session File from an Input File**

Special String in Input File	Translation in Session File			
	ANSI Mode <sup>1</sup>		VT52 Mode <sup>1</sup>	
	Cursor Key Mode Reset Normal	Cursor Key Mode Set Application	Cursor Key Mode Reset Normal	Cursor Key Mode Set Application
{DOWN_ARROW} and {DOWN}	CSI B and ESC [ <sup>3</sup> B	SS3 B and ESC O <sup>2</sup> B	ESC B	ESC B
{RIGHT_ARROW} and {RIGHT}	CSI C and ESC [ <sup>3</sup> C	SS3 C and ESC O <sup>2</sup> C	ESC C	ESC C
{LEFT_ARROW} and {LEFT}	CSI D and ESC [ <sup>3</sup> D	SS3 D and ESC O <sup>2</sup> D	ESC D	ESC D

<sup>1</sup>ANSI mode applies to VT200 and VT100 modes. VT52 mode is an ANSI-incompatible mode.

<sup>2</sup>ESC O is the 7-bit code extension equivalent for the 8-bit control string SS3.

<sup>3</sup>ESC [ is the 7-bit code extension equivalent for the 8-bit control string CSI.

Table 8–9 describes special string translations for codes associated with recording functions when a session file is recorded from an input file.

**Table 8–9: Translation of Special Strings Representing the Recording Functions When Recording a Session File from an Input File**

Special String in Input File	Translation in Session File
{BEGIN_COMPARE}	DLE <sup>1</sup> B
{COMPARE_SCREEN}	DLE <sup>1</sup> C
{END_COMPARE}	DLE <sup>1</sup> E
{DELAY}	DLE <sup>1</sup> D
{WAIT}	DLE <sup>1</sup> W

<sup>1</sup> DLE (CTRL/P) is used here only as an example. The actual value will be the termination character specified when the session file is recorded.

(continued on next page)

**Table 8–9 (Cont.): Translation of Special Strings Representing the Recording Functions When Recording a Session File from an Input File**

<b>Special String in Input File</b>	<b>Translation in Session File</b>
{BEGIN_COMMENT}	DLE <sup>1</sup> !
{END_COMMENT}	SUB (CTRL/Z)

<sup>1</sup> DLE (CTRL/P) is used here only as an example. The actual value will be the termination character specified when the session file is recorded.





# Working with DECwindows Session Files

---

When you record a DECwindows test, DEC/Test Manager creates a binary session file that you can convert to ASCII format using the **EXTRACT** command. To create an ASCII **input file** from a DECwindows session file, pull down the Testing menu and choose the Extract menu item and the DECwindows... submenu item. Then fill in the DECwindows Extract dialog box.

You can also create an ASCII input file from a DECwindows session file using the following command:

```
DTM> EXTRACT DW_TEST.SESSION/DECWINDOWS
```

DEC/Test Manager creates a file called **DW\_TEST.INP**.

You can use an input file to perform the following operations:

- **Make comments** on the various parts of the input file that compose a task
- **Place synchronization points** in the input file
- **Change the time delays** between input events
- **Add looping** to parts of the input file
- **Send informational messages** to a terminal when you play back the session file

---

## 9.1 Creating a DECwindows Input File

A DECwindows session file is a binary file containing machine-readable data. When you use the **EXTRACT** command, the DECwindows session file is translated to ASCII text (the input file). Example 9–1 shows a sample session file that has been translated into an input file.

### NOTE

The mouse motion events in Example 9–1 and subsequent session file examples have been edited to show only the first and last MotionNotify events in a sequence.

#### Example 9–1: DECwindows Input File

---

```
$ DTM EXTRACT DW_TEST.SESSION/DECWINDOWS
.
.
.
$ TYPE DW_TEST.INP
KeyPress      00:00.40 d
KeyRelease   00:00.00 d
KeyPress      00:00.19 i
KeyRelease   00:00.00 i
.ImageText8   di
KeyPress      00:00.22 r
KeyRelease   00:00.00 r
.ImageText8   r
KeyPress      00:00.40 <SPACE>
KeyRelease   00:00.00 <SPACE>
.ImageText8
KeyPress      00:00.28 <SHIFT>
KeyPress      00:00.19 *
KeyRelease   00:00.00 *
.ImageText8   *
KeyRelease   00:00.16 <SHIFT>
KeyPress      00:00.67 .
KeyRelease   00:00.00 .
.ImageText8   .
KeyPress      00:00.07 b
KeyRelease   00:00.00 b
.ImageText8   b
KeyPress      00:00.30 i
KeyRelease   00:00.00 i
.ImageText8   i
KeyPress      00:00.30 n
KeyRelease   00:00.00 n
```

---

(continued on next page)

## Example 9-1 (Cont.): DECwindows Input File

---

```
.ImageText8      n
KeyPress         00:00.24 <SHIFT>
KeyPress         00:00.11 *
KeyRelease      00:00.00 *
.ImageText8      *
KeyRelease      00:00.17 <SHIFT>
KeyPress        00:00.02 <DELETE>
.ImageText8
KeyRelease      00:00.12 <DELETE>
KeyPress        00:00.20 a
KeyRelease      00:00.00 a
.ImageText8      a
KeyPress        00:00.22 r
KeyRelease      00:00.00 r
.ImageText8      r
KeyPress        00:00.27 y
KeyRelease      00:00.00 y
.ImageText8      y
KeyPress        00:00.23 <RETURN>
KeyRelease      00:00.08 <RETURN>
.ImageText8      es found
.ImageText8      Baggit>
MotionNotify    00:02.41      85  105
MotionNotify    00:00.11      134 110
ButtonPress     00:00.28 MB1  134 110
.PolyText8      Commands
.PolyText8      off top
.PolyText8      Display
.PolyText8      e Window
.PolyText8      ications
.PolyText8      Terminal
.PolyText8      Quit
ButtonRelease   00:00.67 MB1  134 110
.PolyText8      Commands
.ImageText8      $ dir
.ImageText8      es found
.ImageText8      $
MotionNotify    00:00.03      133 109
MotionNotify    00:00.28      222 111
ButtonPress     00:00.01 MB1  222 111
.PolyText8      Edit
.PolyText8      Copy
.PolyText8      Paste
.PolyText8      lect All
ButtonRelease   00:00.54 MB1  222 111
.PolyText8      Edit
MotionNotify    00:00.30      223 111
MotionNotify    00:00.17      273 111
ButtonPress     00:00.00 MB1  273 111
.PolyText8      ustomize
.PolyText8      indow...
.PolyText8      splay...
```

---

(continued on next page)

### Example 9-1 (Cont.): DECwindows Input File

---

```
.PolyText8      neral...
.PolyText8      board...
.PolyText8      ction...
.PolyText8      Settings
.PolyText8      Defaults
.PolyText8      from...
.PolyText8      Settings
.PolyText8      gs as...
ButtonRelease   00:01.02 MB1  273  111
.PolyText8      ustomize
```

---

---

## 9.2 Creating a DECwindows Session File from a DECwindows Input File

DEC/Test Manager also enables you to translate an ASCII input file back into session file format using the **RESTORE** command; to do this, pull down the Testing menu, choose the Restore menu item, then choose the DECwindows... submenu item and fill in the DECwindows Restore dialog box.

You can also translate an ASCII input file to a DECwindows session file using the following command:

```
DTM> RESTORE DW_TEST.INP/DECWINDOWS
```

DEC/Test Manager translates the ASCII input file into a binary DECwindows session file called `DW_TEST.SESSION`. If you make an error editing the input file, DEC/Test Manager issues an error message with the line number of the error; a session file is created but it will be incomplete.

You can also create a DECwindows session file with an input file by issuing the DEC/Test Manager **RECORD** command with the `/INPUT` qualifier. In addition to the session file, this will also generate a new benchmark for the test.

---

## 9.3 Editing a DECwindows Input File

Table 9–1 shows the types of editing that you can perform on an input file.

**Table 9–1: Input File Editing Operations**

---

<b>Edit Command</b>	<b>Description</b>
!	An exclamation mark in the first column indicates descriptive text that will not be played back as part of the session file.
*	An asterisk in the first column, replacing a period, pauses the play back mechanism from sending input events to the workstation until a record containing the selected output text has been received from the workstation.
Loop # and EndLoop #	The Loop # and EndLoop # commands indicate where DEC/Test Manager is to repeat the input and output events from the session file. A matched pair of numbers are specified with each loop pair to identify different looping sequences; nested loop sequences are allowed.
SendConsole	The SendConsole command sends text to SYS\$OUTPUT at specified intervals during the play back of a session file.
00:00.00	You can change the times shown in an input file for shorter or longer time periods between input events. Do not change the position of the time stamp or you will cause syntax errors.

---

You also can record several tasks into several session files and then extract them into input files which can be merged.

---

### 9.3.1 Commenting Input Files

Records in an input file that have an exclamation mark (!) in column one are comments. When a session file is created from the input file, the comments will be placed into the session file, however, they will be ignored when the session file is played. Example 9–2 shows a partial input file with comments.

## Example 9-2: Commented Input File

---

```
!  
! This input file executes the DCL DIRECTORY command and then causes MBI to  
! click on the Commands, Edit, and Customize menus of a terminal emulator.  
!  
! Motions have been edited to show only the starting and ending X and Y  
! coordinates.  
!  
KeyPress          00:00.40 d  
KeyRelease        00:00.00 d  
.  
.  
.  
!  
! End  
!
```

---

### 9.3.2 Synchronizing Play Back

Input events can be lost if they are sent to a workstation before a workstation can process the input. For example, if the mouse clicks on a menu item that has not yet appeared on the screen, the application will not be notified of the selection, and any following input will be meaningless.

You can edit an input file to send data to the workstation (or `SYS$OUTPUT`) when the workstation is in a ready state.

You can specify synchronization records that cause the playback system, when it reads the record, to stop sending data to the workstation until the workstation returns specified output text to the playback system. When the output text is received from the workstation, the playback system resumes sending input data to the workstation.

You should add synchronization records where you would normally wait for the application to send data to the workstation. For example, a synchronization record should be placed on the last item of a menu that will be pulled down during play back. When the last menu item is drawn on the workstation, the playback system begins the next mouse motion event.

Output records sent from the workstation have a period (.) in column one. If you change the period to an asterisk (\*), the record becomes a synchronization record for play back.

Data records contain the last eight characters of a text output request string. If the string is less than or equal to eight characters, the whole string is contained in the output record.

Example 9-3 shows a partial input file with synchronization records.

### Example 9-3: Adding Synchronization Points

---

```
.
.
.ImageText8      es found
*ImageText8      Baggit>
MotionNotify    00:02.41      85  105
MotionNotify    00:00.11      134 110
!
! Pop up the Command menu on a terminal emulator
!
ButtonPress     00:00.28 MB1  134 110
.
.
*PolyText8      Quit
ButtonRelease   00:00.67 MB1  134 110
*PolyText8      Commands
.
.
!
! End
!
```

---

A synchronization record must be unique back to (but not including) the previous synchronization record. Consecutive and equal text strings can be synchronization records. However, you cannot have an unmarked output record with the same text string as the previous synchronization record, or a warning message is issued.

---

### 9.3.3 Repeating Tasks in an Input File

You can repeat tasks within an input file using the `Loop` and `EndLoop` commands. For example, to repeat pulling down a menu, place a `Loop` command before a `ButtonPress` record and place an `EndLoop` command after the `ButtonRelease` and the last menu item in the menu.

`Loop` and `EndLoop` commands must begin in the second column. The pair must be assigned a number, which specifies the number of times to repeat the task. You can nest `Loop` and `EndLoop` commands. The `Loop` and `EndLoop` command keywords are case sensitive.

Example 9-4 shows an input file that repeats the entire sequence five times and repeats pulling down the Commands menu twice.

#### Example 9-4: Adding Loops to an Input File

---

```
!  
! This input file executes the DCL DIRECTORY command and then causes MB1  
! click on the Commands, Edit, and Customize menus of a terminal emulator  
!  
! Motions have been edited to show only the starting and ending X and Y  
! coordinates.  
!  
! Repeat this input file five times.  
!  
Loop 5  
KeyPress          00:00.40 d  
KeyRelease        00:00.00 d  
.  
.  
.  
MotionNotify      00:02.41      85 105  
MotionNotify      00:00.11      134 110  
!  
! Pop up the Commands menu on a terminal emulator  
! Repeat this task twice.  
!  
Loop 2  
ButtonPress        00:00.28 MB1 134 110  
.PolyText8         Commands  
.PolyText8         off top  
.PolyText8         Display  
.PolyText8         e Window  
.PolyText8         ications  
.PolyText8         Terminal  
*PolyText8         Quit  
ButtonRelease      00:00.67 MB1 134 110  
*PolyText8         Commands  
.ImageText8        $ dir  
.ImageText8        es found  
.ImageText8        $  
EndLoop 2  
!  
! End Commands menu  
!  
.  
.  
.  
EndLoop 5  
!  
! End  
!
```

---



---

## 9.3.4 Creating Informational Messages

You can place informational messages in an input file. When converted to a session file and played back, these messages can be sent to SYSS\$OUTPUT. An informational message does not affect the session file and can help by identifying processing points in a session file.

To create an informational message, specify the SendConsole command beginning in column two with a following message. The SendConsole command is case sensitive.

Example 9–5 shows the entire input file with informational messages and all previous edits.

### Example 9–5: Creating Informational Messages

---

```
!  
! This input file executes the DCL DIRECTORY command and then causes MB1 to  
! click on the Commands, Edit, and Customize menus of a terminal emulator.  
!  
! Motions have been edited to show only the starting and ending X and Y  
! coordinates.  
!  
! Repeat this input file five times.  
!  
Loop 5  
SendConsole      ...Starting main loop...  
KeyPress         00:00.40 d  
KeyRelease       00:00.00 d  
KeyPress         00:00.19 i  
KeyRelease       00:00.00 i  
.ImageText8      di  
KeyPress         00:00.22 r  
KeyRelease       00:00.00 r  
.ImageText8      r  
KeyPress         00:00.40 <SPACE>  
KeyRelease       00:00.00 <SPACE>  
.ImageText8  
KeyPress         00:00.28 <SHIFT>  
KeyPress         00:00.19 *  
KeyRelease       00:00.00 *  
.ImageText8      *  
KeyRelease       00:00.16 <SHIFT>  
KeyPress         00:00.67 .  
KeyRelease       00:00.00 .  
.ImageText8      .  
KeyPress         00:00.07 b  
KeyRelease       00:00.00 b
```

---

(continued on next page)

## Example 9-5 (Cont.): Creating Informational Messages

```
.ImageText8      b
KeyPress         00:00.30 i
KeyRelease      00:00.00 i
.ImageText8      i
KeyPress         00:00.30 n
KeyRelease      00:00.00 n
.ImageText8      n
KeyPress         00:00.24 <SHIFT>
KeyPress         00:00.11 *
KeyRelease      00:00.00 *
.ImageText8      *
KeyRelease      00:00.17 <SHIFT>
KeyPress         00:00.02 <DELETE>
.ImageText8      <DELETE>
KeyRelease      00:00.12 <DELETE>
KeyPress         00:00.20 a
KeyRelease      00:00.00 a
.ImageText8      a
KeyPress         00:00.22 r
KeyRelease      00:00.00 r
.ImageText8      r
KeyPress         00:00.27 y
KeyRelease      00:00.00 y
.ImageText8      y
KeyPress         00:00.23 <RETURN>
KeyRelease      00:00.08 <RETURN>
.ImageText8      es found
*ImageText8     Baggit>
MotionNotify    00:02.41      85 105
MotionNotify    00:00.11     134 110
!
! Pop up the Commands menu on a terminal emulator
! Repeat this task twice.
!
Loop 2
SendConsole     ...Commands menu loop...
ButtonPress     00:00.28 MB1 134 110
.PolyText8     Commands
.PolyText8     off top
.PolyText8     Display
.PolyText8     e Window
.PolyText8     ications
.PolyText8     Terminal
*PolyText8     Quit
ButtonRelease   00:00.67 MB1 134 110
*PolyText8     Commands
.ImageText8     $ dir
.ImageText8     es found
.ImageText8     $
EndLoop 2
!
! End Commands menu
```

(continued on next pag

## Example 9-5 (Cont.): Creating Informational Messages

---

```
!  
! Pop up the Edit menu on a terminal emulator  
!  
MotionNotify      00:00.03      133  109  
MotionNotify      00:00.28      222  111  
SendConsole        ...Edit Menu...  
ButtonPress        00:00.01 MB1  222  111  
.PolyText8         Edit  
.PolyText8         Copy  
.PolyText8         Paste  
*PolyText8         lect All  
  ButtonRelease    00:00.54 MB1  222  111  
*PolyText8         Edit  
!  
! End Edit menu  
!  
! Pop up the Customize menu on a terminal emulator  
!  
MotionNotify      00:00.30      223  111  
MotionNotify      00:00.17      273  111  
SendConsole        ...Customize Menu...  
ButtonPress        00:00.00 MB1  273  111  
.PolyText8         ustomize  
.PolyText8         indow...  
.PolyText8         splay...  
.PolyText8         neral...  
.PolyText8         board...  
.PolyText8         ction...  
.PolyText8         Settings  
.PolyText8         Defaults  
.PolyText8         from...  
.PolyText8         Settings  
*PolyText8         gs as...  
  ButtonRelease    00:01.02 MB1  273  111  
*PolyText8         ustomize  
!  
! End Customize menu  
!  
EndLoop 5  
!  
! End input  
!  
SendConsole        ...input file Complete...  
!
```

---

### 9.3.5 Changing the Times of Input Events

You can change the times shown in an input file for shorter or longer time periods between input events by editing the time stamps. Do not change the position of the time stamp.



## DEC/Test Manager Callable Interface

---

This chapter describes the callable interface for DEC/Test Manager. DTM\$DTM is a high-level entry point that enables calling programs to pass a DCL command line to DEC/Test Manager for processing. DTM\$DTM parses and executes the command line, and then returns to the calling program. It can return all DEC/Test Manager return codes and CLI\$ errors. The DTM\$DTM routine provides a full command-line level interface into DEC/Test Manager.

---

### 10.1 Calling Sequence for DTM\$DTM

The format for using DTM\$DTM is as follows:

```
DTM$DTM( [command_line],  
         [msg_routine],  
         [prompt_routine],  
         [confirm_routine],  
         [output_routine],  
         [width],  
         [init_flag])
```

To perform confirmations, prompting, or display output, you must supply callback routines. The following sections describe these callback routines and other parameters to the DTM\$DTM routine.

---

### 10.1.1 Command Line (`command_line`)

Type: `char_string`  
Access: `read`  
Mechanism: `by descriptor`

The command line parameter specifies the address of a string descriptor that contains a command line. If you specify a value of 0, DEC/Test Manager calls the `prompt_routine` if specified to obtain a command line. If you do not specify this argument or a prompt routine, DEC/Test Manager returns the error `RMS$_EOF` (end of file detected).

---

### 10.1.2 Message Routine (`msg_routine`)

Type: `procedure`  
Access: `read`  
Mechanism: `by reference`

The message routine specifies a message handler routine. See Section 10.4 for information about writing a message handler routine.

---

### 10.1.3 Prompt Routine (`prompt_routine`)

Type: `procedure`  
Access: `read`  
Mechanism: `by reference`

The prompt routine parameter specifies the address of a callback routine that is called when the caller specifies a missing or incomplete line.

If you do not specify a prompt callback, DEC/Test Manager does not prompt you, but operates as if a callback had been specified and had returned the status `RMS$_EOF` (except in the case of prompting for a DEC/Test Manager remark, where the status is `RMS$_NORMAL`). The `RMS$_EOF` return status causes termination of command parsing (as if the user had pressed CTRL/Z at the DCL prompt).

The prompt callback routine is called with three parameters:

**response\_string**

Specifies the address of a descriptor that points to the character string into which the prompt routine writes the text in response to the prompt.

**prompt\_string**

Specifies a string descriptor containing the prompt string, passed by reference.

**response\_string\_length**

Specifies the number of bytes written into response-string by the prompt routine.

**NOTE**

The parameters you specify for the prompt routine parallel the parameters for the Run Time Library (RTL) routine, LIB\$GET\_INPUT, allowing LIB\$GET\_INPUT to be specified as your callback routine.

---

**10.1.4 Confirmation Routine (confirm\_routine)**

Type: procedure  
Access: read  
Mechanism: by reference

When the /CONFIRM qualifier is specified as part of the command line, the confirmation routine parameter specifies the address of a callback routine that is used, rather than specifying direct terminal input.

This routine works in either of two modes. It may return a string in the response string parameter or the status of whatever operation it used to obtain the string (for example, LIB\$GET\_INPUT or \$QIO status). Table 10-1 describes the valid values that may be returned in the response string.

**Table 10–1: Confirm\_Routine Response String**

<b>String</b>	<b>Meaning</b>
YES, 1, true	Indicates positive confirmation
ALL	Indicates positive confirmation and that future actions of the current call to DEC/Test Manager should be carried out without confirmation
NO, 0, false	Indicates negative confirmation
QUIT	Indicates negative confirmation and that DEC/Test Manager performs no further actions

The confirmation routine may also return a DEC/Test Manager confirmation status code as in Table 10–2:

**Table 10–2: Confirm Routine Return Status**

<b>Return Code</b>	<b>Meaning</b>
DTM\$_CONFIRM	Yes
DTM\$_NOCONFIRM	No
DTM\$_ALL	All
DTM\$_STOPPED	Quit

If the callback routine returns one of these codes, then any string returned is ignored. The callback routine should return a status of DTM\$\_NORMAL when returning a response string.

For confirmations where ALL and QUIT are not meaningful, ALL is equivalent to YES and QUIT is equivalent to NO.

If you do not specify a confirm callback routine, DEC/Test Manager does not request confirmation. It operates as if a callback had been specified and had returned the string YES. DEC/Test Manager then proceeds with the operation.

If an invalid response is given, DEC/Test Manager prompts you again. Note that any response can be abbreviated to a single character. If a null string is returned, DEC/Test Manager defaults to NO.

The confirm callback routine is called with two parameters:



**response\_string**

Specifies the address of a descriptor that points to the character string into which the confirm routine writes the text in response to the confirmation prompt.

**prompt\_string**

Specifies a string descriptor passed by reference for the prompt string, which can then be displayed to the user.

---

## 10.1.5 Output Routine (**output\_routine**)

Type: procedure  
Access: read  
Mechanism: by reference

The output routine parameter specifies the address of a callback routine to handle output usually sent to SYS\$OUTPUT. For example, all output from a SHOW command is directed to SYS\$OUTPUT (in the absence of an overriding /OUTPUT qualifier). Note also that if you specify the /OUTPUT qualifier to redirect terminal output to a file, DEC/Test Manager opens, writes to, and closes the file normally and does not use the output callback routine. This callback also receives the output for the commands specifying the /OUTPUT=SYS\$OUTPUT qualifier.

If you do not specify output\_routine, DEC/Test Manager writes all output to SYS\$OUTPUT.

The output callback routine is called with one parameter:

**output\_string**

Specifies a string descriptor for the output string, passed by reference.

---

## 10.1.6 Output Width (**width**)

Type: longword\_signed  
Access: read  
Mechanism: by reference

The output width parameter specifies the maximum width of text that can be sent to the output callback routine. If you do not specify this argument, or you specify a value of 0 or less, the terminal width is used. If this is unavailable, the width defaults to 132 characters.

---

## 10.1.7 Initialization Flag (`init_flag`)

Type: longword\_signed

Access: read

Mechanism: by reference

The initialization flag parameter specifies whether DEC/Test Manager processes an existing DEC/Test Manager initialization command file before processing the passed command line. If this argument is not specified, the default is to execute the initialization file. Specifying a 0 value suppresses the execution of the initialization command file.

---

## 10.2 Rules for Writing DEC/Test Manager Callback Routines

The following list describes rules to follow when you write DEC/Test Manager callback routines:

- Every callback routine must return control to DEC/Test Manager. If your routine does not return control to DEC/Test Manager, DEC/Test Manager cannot finish the transaction and the library remains locked. (If your library becomes locked, you must use the `VERIFY` command with the `/RECOVER` qualifier to unlock it.) In addition, any resources used to process the command are not released.
- Callback routines must return a defined condition value to DEC/Test Manager. You can use `DTM$NORMAL` to indicate successful completion of the callback routine, or you can return a condition code from a VMS system service or other system software. DEC/Test Manager checks for the `DTM$EOF` and `RMS$EOF` values, and also checks the low-order bit to determine if the status code indicates success.
- A success code directs DEC/Test Manager to continue processing. If more data awaits processing, DEC/Test Manager calls the callback routine again.
- If the callback routine encounters an error during processing, it aborts the DEC/Test Manager call by returning an error status. This causes the DEC/Test Manager call to exit.

---

## 10.3 Handling Error Conditions

DEC/Test Manager handles error conditions in one of two ways:

- If the condition is not fatal, DEC/Test Manager calls a message handler. You can provide a message routine to handle messages (see Section 10.4), or, if you do not provide a message routine, DEC/Test Manager calls its own message handler.
- If the condition is fatal, DEC/Test Manager signals the error. Fatal conditions are those situations where execution cannot continue. DEC/Test Manager does not call the message routine under these circumstances.

If you have established a condition handler in the calling program and the condition handler encounters a fatal return value, do not return a value of `SS$_CONTINUE` from the condition handler or signal `SS$_CONTINUE` again, and do not issue additional calls to DEC/Test Manager until you have exited and entered the image again. The fatal error indicates that DEC/Test Manager cannot continue with the current invocation of the image.

If you supply a routine for input or output and you establish a condition handler within this routine, do not exit from the image (through either the condition handler or the routine itself).

To exit the image, you should return an error (any status with the low bit clear) from your routine, causing DEC/Test Manager to terminate with `DTM$_USERERR` status. `DTM$_USERERR` status indicates that a callback routine returned an error.

---

## 10.4 Writing an Error Message Handler

DEC/Test Manager directs all diagnostic messages to the default destinations `SYS$OUTPUT` and `SYS$ERROR`. However, you can write your own routine to handle messages. When you specify the `msg_routine` parameter to the `DTM$DTM` routine, DEC/Test Manager passes control to your message handler instead of using the default handler. DEC/Test Manager does not call your message handler routine if a fatal condition occurs, but instead notifies you by signaling the condition. If you receive a fatal error message, you should exit and enter DEC/Test Manager again; do not attempt to recall DEC/Test Manager within the same image invocation if DEC/Test Manager detected a fatal error.

DEC/Test Manager passes the following parameters in the order shown with each call to `msg_routine`:

(`signal_array`, `mechanism_array`)

### **signal\_array**

Type: `vector_longword_unsigned`

Access: `read`

Mechanism: `by reference`

Specifies a standard VMS signal array.

### **mechanism\_array**

Type: `vector_longword_unsigned`

Access: `read`

Mechanism: `by reference`

Specifies a standard VMS mechanism array.

The following list describes rules to follow when you write message-handling routines:

- Do not invoke any DEC/Test Manager routines from a message routine.
- Do not use the `LIB$ESTABLISH` Run-Time Library routine to enable the message routine as the exception handler for a DEC/Test Manager call. DEC/Test Manager uses its own exception handlers and calls the user-supplied message routine under the correct circumstances. (The message routine is only for handling messages, not for general exception handling during the execution of a DEC/Test Manager routine.)

---

## **10.5 Linking with the DEC/Test Manager Image**

You need to specify the DEC/Test Manager shareable image to the `DCL LINK` command. You explicitly reference the DEC/Test Manager shareable image (`SYS$SHARE:DTMSHR.EXE`) by specifying the linker option as follows:

```
$ LINK filename[,...],SYS$INPUT/OPTIONS  
SYS$SHARE:DTMSHR.EXE/SHARE  
CTRL/Z
```

# Command Dictionary

---

The Command Dictionary describes the elements of the DEC/Test Manager command line and defines the syntax rules for entering commands. It also describes each DEC/Test Manager command.



---

# 1 Command Format

DEC/Test Manager commands have the following format:

DTM> command [parameter...] [/qualifier...] "remark"

Table CD-1 describes the command line elements.

**Table CD-1: DEC/Test Manager Command Line Elements**

Element	Description
Command	Describes the DEC/Test Manager action. Commands are a required part of the DEC/Test Manager command line.
Parameter	Specifies information required by some commands. Refer to the specific commands in this dictionary to determine whether a command requires parameters.
/Qualifier	Modifies the DEC/Test Manager action in a specific way. Qualifiers are an optional part of the DEC/Test Manager command line and can be placed anywhere on the command line after the command, in any combination. One exception to this occurs with test group expression qualifiers, which you must specify immediately following the test group expression to which it refers.
Remark	Associates a comment with any library-changing command to be logged in the history log file. Remarks are required on library-changing commands, although null remarks are acceptable.

The following sections describe the formats for the DEC/Test Manager commands, parameters, qualifiers, and parameter qualifiers.

You can abbreviate DEC/Test Manager commands by specifying the minimum number of characters that uniquely identifies the command.

---

## 1.1 Command Parameters

Parameter values for the DEC/Test Manager commands consist of the following:

- Collection names and expressions
- Group names and expressions
- Test names and expressions

- Variable names and expressions
- Variable values
- Result description names and expressions
- Object expressions
- Test group expressions
- File names

You can use the same name for a collection, group, test description, or variable. You cannot use names beginning with DTM\$, nor can you use the variable names P1 through P8. These names are reserved for use by DEC/Test Manager.

A name refers to a single item such as a collection, group, test, variable, and result description.

An expression refers to one or more items such as collections, groups, tests, variables, result descriptions, objects, and test groups.

Collection expressions, group expressions, result description expressions, test expressions, and variable expressions all contain names of the same type. For example, a collection expression contains only collection names and expressions that resolve to collection names.

#### **NOTE**

You cannot list items in result description expressions.

Object expressions and test group expressions contain more than one type of name. Object expressions can contain test names, group names, and collection names. Test group expressions can contain only test names and group names.

Expressions permit the use of wildcards. Use a comma to separate items in a list.

---

## **1.2 Qualifiers**

DEC/Test Manager uses command qualifiers and parameter qualifiers. The following sections describe both types of qualifiers and their uses.



---

### 1.2.1 Command Qualifiers

Command qualifiers modify the command; you can place them anywhere on the command line after the command and in any combination. You can enter command qualifiers before or after any parameters and before or after the remark. Some command qualifiers require a value.

If input is taken from a command file or if the command is issued in batch, the action is performed without confirmation, regardless of whether you specified the /CONFIRM or /NOCONFIRM qualifier.

---

### 1.2.2 Parameter Qualifiers

The two parameter qualifiers are /GROUP and /TEST\_DESCRIPTION.

You use parameter qualifiers only with test group expressions to differentiate the items contained in the test group expression as representing either tests or groups.

The position of these qualifiers is significant. The /GROUP parameter qualifier identifies the item that it follows as a group name. Similarly, the /TEST\_DESCRIPTION parameter qualifier identifies as a test name the item that it follows. The default parameter qualifier is /TEST\_DESCRIPTION. Thus, if no parameter qualifier follows an item in a test group expression, DEC/Test Manager identifies that item as a test name. For example:

```
LMTEST1,LMTESTS/GROUP,RMTEST
```

In this test group expression, DEC/Test Manager identifies LMTEST1 and RMTEST as test names and LMTESTS as a group name.

---

### 1.3 Remark

A **remark** is a comment that is associated with any command that modifies the DEC/Test Manager library. A remark can consist of up to 255 printable ASCII characters; the total command line length follows VMS conventions, so remarks are truncated if necessary. If a remark includes any space characters, you must enclose the remark in quotation marks (" "). A remark is required with any library-modifying commands, but the remark can be null. Remarks are stored in the DEC/Test Manager history log file.

If you do not specify a remark on a library-modifying command, DEC/Test Manager prompts you for one. You do not need to include quotation marks with these remarks.

The remark parameter specifies the remark logged with the COPY or MODIFY command in the history file. The remark qualifier specifies the remark associated with the modified or copied group, test description, or variable.

You can also add a remark to the history by using the REMARK command. This command enables you to log any comments, not just usual events.

---

## 2 File Specification Format

Whenever you specify a file for use in DEC/Test Manager, you must use a standard VMS file specification. For a complete description of a file specification, see the *VMS DCL Concepts Manual*. The format for a file specification is as follows:

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

---

## 3 Command Descriptions

The commands in this section are arranged in alphabetical order with each command description containing the following:

- Command Format
- Restrictions, if any
- Command parameters
- Descriptions of the command
- Command qualifiers (defaults, if any, are marked (D))
- Parameter qualifiers, if any (defaults, if any, are marked (D))
- Examples

The REVIEW command places DEC/Test Manager at a subsystem level. The commands used at this level are discussed in Section 4. The REVIEW subcommands are documented with the same format as the other DEC/Test Manager commands.

---

## @file-specification

Executes DEC/Test Manager commands contained in the specified file.

---

### Format

**@file-specification**

Command Qualifiers	Defaults
None	None

---

### Command Parameter

***file-specification***

Specifies the command procedure to execute. If the file specification does not include a file type, DEC/Test Manager uses the default file type .COM.

---

### Description

The @file-specification command executes the commands in the specified file. The file can contain any DEC/Test Manager command, including another @file-specification command.

When DEC/Test Manager executes an EXIT command or reaches the end of the command procedure, control is returned to the current command level. The invoking command stream can be either the terminal or another command procedure.

Do not preface the commands in the specified file with the DTM command or dollar sign (\$). For example, enter SHOW LIBRARY, not DTM SHOW LIBRARY or \$ SHOW LIBRARY.

## @file-specification

---

### Example

```
DTM> @MAIL_TEST
%DTM-S-LIBIS, DEC/Test Manager library is DUA0:[USER01.DTMLIB]
%DTM-S-RESUBMITTED, collection MAIL_TEST has been resubmitted
-DTM-I-TEXT, Job MAIL_TEST (queue SYS$BATCH, entry 18) started on SYS$BATCI
DTM>
```

This example executes the command procedure MAIL\_TEST.COM containing the commands SET LIBRARY DUA0:[USER01.DTMLIB] and SUBMIT MAIL\_TEST.

---

# ATTACH

Switches control from your current process to another process in your job.

---

## Format

**ATTACH** [*process-name*] [*/qualifier*]

Command Qualifiers	Defaults
/IDENTIFICATION=pid	None
/PARENT	None

---

## Command Parameter

### *process-name*

Specifies an existing process to which you want to attach your terminal.

If you specify either the /IDENTIFICATION or /PARENT qualifier, do not specify the process name parameter or the other qualifier. If you do not specify a qualifier, you must specify a process name.

---

## Description

The ATTACH command enables you to change control to a subprocesses created with the SPAWN command or to reconnect to a parent (original) process.

You can use the ATTACH command in conjunction with the SPAWN/WAIT command to return to a DEC/Test Manager session without terminating the subprocess. See the SPAWN command for more information.

# ATTACH

---

## Command Qualifiers

### ***/IDENTIFICATION=pid***

Specifies the process identification (PID) of the process to which you want to attach your terminal. You can omit the leading zeros when you specify a PID.

If you specify the */IDENTIFICATION* qualifier, do not specify the process name parameter or the */PARENT* qualifier. If you do not specify a qualifier you must specify a process name.

### ***/PARENT***

Specifies that the process you want to attach to is your original (parent) process.

If you specify the */PARENT* qualifier, do not specify the process name parameter or the */IDENTIFICATION* qualifier. If you do not specify a qualifier, you must specify a process name.

---

## Example

```
MAIL> SPAWN DTM
.
.
.
DTM> ATTACH/PARENT
You have 0 new messages.

MAIL>
```

This example uses the VMS Mail Utility (MAIL) command SPAWN to create a subprocess running DEC/Test Manager. The DEC/Test Manager ATTACH command is then used to attach the terminal back to the MAIL session, the parent process.

---

# COMPARE

Compares the result file produced for each test description in a collection with its corresponding benchmark file.

---

## Format

**COMPARE** *collection-name* [/qualifier...]

### Command Qualifiers

/CHARACTERS

/FULL

/IGNORE=(keyword,...)

/[NO]LOG

/[NO]PARALLEL

/RECORDS

/SCREENS

/SENTINEL=("begin-delimiter", "end-delimiter")

/WIDTH=n

### Defaults

See text

None

None

/LOG

/NOPARALLEL

/RECORDS (for noninteractive tests)

/SCREENS (for interactive tests)

None

/WIDTH=132

---

## Command Parameter

### *collection-name*

Specifies a name that identifies a collection of test descriptions to be compared. A collection name consists of up to 39 characters. You must specify a collection name; you cannot specify a collection expression containing wildcard characters or a list.

---

## Description

The COMPARE command prepares a collection to be reviewed by comparing the results generated for each test that was run to the test's benchmark file. This command works only with a collection that has been run but not reviewed.

# COMPARE

The COMPARE command compares the results of interactive terminal and DECwindows tests, and noninteractive tests. You can compare interactive tests in three ways:

- Screen by screen using the /SCREENS qualifier
- Record by record using the /RECORDS qualifier (noninteractive and terminal tests only)
- Character by character using the /CHARACTERS qualifier (noninteractive and terminal tests only)

When you perform the comparison character by character, record boundaries in the result file are ignored. Line feed and escape characters are used to break the output into lines, which are then processed for differences.

DEC/Test Manager informs you if you use the COMPARE command with a collection that has already been compared. DEC/Test Manager automatically compares collections when they are executed. To prevent a collection from being automatically compared, specify the /NOCOMPARE qualifier with the CREATE COLLECTION command. You cannot use the COMPARE command with a collection that is in use.

The COMPARE command compares the completed part of a partially run collection. If you have a collection that does not run to completion, you can compare and review the tests in the collection that did run. A partially run collection results if the system crashes while the collection is executing, if you terminate the RUN command by pressing CTRL/C, or if you stop execution of the collection with the STOP command.

The following table shows the results, comparison statuses, and file statuses that can occur for a collection as a result of issuing the COMPARE commands.

<b>Result</b>	<b>Comparison Status</b>	<b>File Status</b>
No Differences	Successful	Result file deleted
Differences	Unsuccessful	Result file created, difference file created



---

<b>Result</b>	<b>Comparison Status</b>	<b>File Status</b>
No Benchmark	New Test	You can create a benchmark file for this test while reviewing the test results from the Review subsystem with the Review subsystem UPDATE command.

---

If you store your benchmark files in a CMS library, DEC/Test Manager searches the CMS library for your benchmark files. If DEC/Test Manager finds that a result file has not been generated for a test, it marks the test as not run.

If an error occurs while a comparison is being performed, the test being compared is given the comparison status of comparison aborted. The comparison status for a test is included in the test's result description.

---

## Command Qualifiers

### ***/CHARACTERS***

Performs a character-by-character comparison of the results file with the benchmark file.

The default is /SCREENS for interactive tests and /RECORDS for noninteractive tests.

### ***/FULL***

For noninteractive and interactive terminal tests, the /FULL qualifier includes a complete listing of the text in the difference file that was identical and a listing of the differences encountered when the result file and benchmark file are compared.

### ***/IGNORE=keyword***

The /IGNORE qualifier enables you to specify that various aspects of benchmark and result files are to be ignored during comparison.

# COMPARE

The following keywords apply to noninteractive and interactive terminal tests, only.

<b>Keyword</b>	<b>Result</b>
CASE	Ignores any differences between the case of alphabetic characters (A,a,B,b, . . . )
FORM-FEEDS	Ignores form-feed characters
LEADING_BLANKS	Ignores leading blanks and tabs
SPACING	Treats multiple blanks and tabs as a single space
TRAILING_BLANKS	Ignores trailing blanks and tabs

For interactive terminal tests, if you specify the `/IGNORE` and `/SCREENS` qualifiers together, DEC/Test Manager performs the comparison screen by screen and ignores the `/IGNORE` qualifier.

If you specify more than one keyword, separate the keywords with commas and enclose the list in parentheses. The output file (your result file) is not changed in any way by the `/IGNORE` qualifier.

The following keyword applies to DECwindows tests only.

<b>Keyword</b>	<b>Result</b>
MASK	Ignores masked areas defined on DECwindows benchmark images

**`/LOG (D)`**

**`/NOLOG`**

Controls whether DEC/Test Manager displays informational and success messages on your screen.

**`/PARALLEL`**

**`/NOPARALLEL (D)`**

Specifies whether the lines that do not match in the result and benchmark files are formatted side by side.

If you specify the `/[NO]PARALLEL` and `/SCREENS` qualifiers together, DEC/Test Manager performs the comparison screen by screen and ignores the `/[NO]PARALLEL` qualifier.

## ***/RECORDS***

For noninteractive and interactive terminal tests, the ***/RECORDS*** qualifier performs a record-by-record comparison of the result and benchmark files. The default is ***/SCREENS*** for interactive tests and ***/RECORDS*** for noninteractive tests.

Records are identical only if they contain the same characters. Use this type of comparison only when you expect the record in which a string appears to be the same each time a comparison is performed.

Use caution when specifying the ***/RECORDS*** qualifier for an interactive test because the records in the result file are not guaranteed to be written the same way each time the test runs. You might want to use the ***/RECORDS*** qualifier for an interactive test whose result file is not generated by DEC/Test Manager, for example, if you rename a test output file to be ***DTM\$RESULT***.

## ***/SCREENS***

Performs a screen-by-screen comparison of the result and benchmark files for an interactive test. The default is ***/SCREENS*** for interactive tests and ***/RECORDS*** for noninteractive tests.

## ***/SENTINEL=("begin-delimiter", "end-delimiter")***

Specifies a pair of strings used to delimit a section of text to be ignored during the comparison of result and benchmark files for noninteractive tests. The delimiters can be up to 256 characters per line, and must be unique. Any text between and including the delimiters is treated as if it did not exist.

If you do not enclose the sentinel strings in quotation marks, they are converted to uppercase before the comparison of the files. Sentinel strings may contain any characters, but if you include spaces or tabs, they must be enclosed in quotation marks.

## ***/WIDTH=n***

For noninteractive and interactive terminal tests that were compared with the ***/CHARACTERS*** or ***/RECORDS*** qualifier, the ***/WIDTH*** qualifier specifies the maximum width allowed for the differences report. The minimum width is 48 columns and the maximum width is 511 columns. The default value is 132 columns.

# COMPARE

---

## Example

```
DTM> COMPARE MAIL_COLL
%DTM-I-SUCCEDED, the comparison for the test MAIL_TEST succeeded
%DTM-I-SUCCEDED, the comparison for the test SEND_MAIL_TEST succeeded
%DTM-S-COMPARED, collection MAIL_COLL compared
```

This example compares the results for all tests in the collection **MAIL\_COLL**. For each test, DEC/Test Manager deletes the result files for tests whose benchmark and result files match, and it creates a difference file for tests whose benchmark and result files differ.

---

## CONVERT LIBRARY

Converts DEC/Test Manager libraries created with a version of DEC/Test Manager prior to Version 2.0 for use with the current version of DEC/Test Manager.

---

### Format

**CONVERT LIBRARY** *existing-library-name new-library-name*

**Command Qualifiers**

None

**Defaults**

None

---

### Command Parameters

***existing-library-name***

Specifies the directory for the existing DEC/Test Manager library you want to convert.

***new-library-name***

Specifies the directory for the new DEC/Test Manager library you want to create.

---

### Description

The **CONVERT LIBRARY** command creates a copy of an existing DEC/Test Manager library and converts the copy for use with this version of DEC/Test Manager. Libraries created with DEC/Test Manager Version 2.0 or later do not need to be converted. Conversion maintains everything in your existing library except collections. The existing library is not altered.

Before converting a library, first create an empty directory to contain the new, converted library.

# CONVERT LIBRARY

---

## Example

```
$ CREATE/DIRECTORY [project.v1lib]
$ DTM
DTM> CONVERT LIBRARY [project.V1lib] [project.v2lib]
%DTM-S-COPIED, V1 variables copied
%DTM-S-COPIED, V1 groups copied
%DTM-S-COPIED, V1 test descriptions copied
-DTM-S-CONVERTED, your V1 library has been successfully converted to V2
```

This example first creates a new directory to contain the converted library. Then the DEC/Test Manager system is entered and the existing library is converted. The benchmark files are not copied because they are stored outside the DEC/Test Manager library in a benchmark directory. They are accessible to the new library.

---

## COPY TEST\_DESCRIPTION

Copies an existing test description.

---

### Format

**COPY TEST\_DESCRIPTION** *test-name1 test-name2 [/qualifier...]*  
*"remark"*

#### Command Qualifiers

/[NO]COMMAND=command  
 /COMPARISON\_TYPE=keyword  
 /[NO]EPILOGUE=file-specification  
 /NOFILTERS  
 /NOGROUPS  
 /[NO]LOG  
 /[NO]PROLOGUE=file-specification  
 /[NO]REMARK="remark"  
 /[NO]TEMPLATE=file-specification  
 /NOVARIABLES

#### Defaults

Current command  
 Current default  
 Current test epilogue  
 Current filters  
 Current group memberships  
 /LOG  
 Current test prologue  
 Current remark  
 Current template  
 Current variables

---

### Restrictions

- The /COMMAND qualifier applies to DECwindows tests only.
- The /NOFILTERS qualifier applies to interactive and noninteractive terminal tests only.

---

### Command Parameters

#### *test-name1*

Specifies the name of the test description to be copied. You cannot use wildcards to specify the test name parameters. The *test-name1* and *test-name2* parameters must be different; you cannot copy a test description to itself.

# COPY TEST\_DESCRIPTION

## ***test-name2***

Specifies the name of the test description to be created. You cannot use wildcards to specify the test name parameters. The *test-name1* and *test-name2* parameters must be different.

## ***remark***

Specifies a string that contains a comment. You must specify a remark within quotation marks (" "); the exception to this rule is when you specify remark string at the remark prompt. If you do not provide a remark string you are prompted for one, however, a null remark is permitted. This remark is associated with the COPY TEST\_DESCRIPTION command and is logged with it in the history file.

---

## Description

The COPY TEST\_DESCRIPTION command makes a copy of an existing test description. This enables you to create several similar test descriptions without entering information into all the test description fields. You can copy a test description as it is, or you can modify test description fields by specifying qualifier values. Because DEC/Test Manager does not permit you to have two test descriptions with the same name, the name for the new test description must be unique.

When you copy a test description, only the information in the fields you specify with command qualifiers is modified; information in the remaining test description fields is copied as is from the existing test description.

The new test description belongs to the same groups as the existing test description; it is associated with the same variables as the existing test description. If the test description you are copying describes an interactive test, the new test is marked interactive.

---

## Command Qualifiers

### ***/COMMAND=command***

By default, the new test description has the same command as the copied test description. If you specify the /COMMAND qualifier, you associate a new command with the new test description. If you specify the



## COPY TEST\_DESCRIPTION

**/NOCOMMAND** qualifier, the associated command is not copied with the test description. The qualifier applies to DECwindows tests only.

### ***/COMPARISON\_TYPE=keyword***

Specifies how the result and benchmark files are to be compared. A comparison type is not associated with the test description. The valid values for keyword are as follows:

<b>Keyword</b>	<b>Meaning</b>
CHARACTERS	Compares files character by character.
RECORDS	Compares files record by record. This is the default for noninteractive terminal tests.
SCREENS	Compares files screen by screen; screens not marked are not compared. This is the default for interactive terminal and DECwindows tests.

DECwindows tests can only use the SCREENS comparison type. The SCREENS comparison type is also the default comparison type for interactive terminal tests. If you specify the **/COMPARISON\_TYPE=SCREENS** qualifier for a noninteractive test, this value is ignored.

### ***/EPILOGUE=file-specification***

#### ***/NOEPILOGUE***

Determines whether a test epilogue file is associated with the test description. The epilogue file associated with the existing test description is also associated with the new test description.

The **/EPILOGUE** qualifier causes the specified epilogue file to replace the existing epilogue file. The **/NOEPILOGUE** qualifier specifies that no epilogue file be associated with the new test description.

### ***/NOFILTERS***

Specifies that no filters be associated with the new test description. By default, the filters associated with the existing interactive or noninteractive terminal test descriptions are also associated with the new test descriptions.

# COPY TEST\_DESCRIPTION

## ***/NOGROUPS***

Specifies that the new test description does not belong to any groups. If you do not specify this qualifier, the new test description belongs to the same group (or groups) as the previous test description.

## ***/LOG (D)***

## ***/NOLOG***

Controls whether DEC/Test Manager displays informational and success messages on your screen.

## ***/PROLOGUE=file-specification***

## ***/NOPROLOGUE***

Determines whether a test prologue file is associated with the test description. The prologue file associated with the existing test description is also associated with the new test description.

The */PROLOGUE* qualifier causes the specified prologue file to replace the existing prologue file. The */NOPROLOGUE* qualifier specifies that no prologue file be associated with the new test description.

## ***/REMARK="string"***

## ***/NOREMARK***

Determines whether a remark is associated with the new test description. By default, the remark associated with the new test description will be a copy of the remark associated with the existing test description. This remark is associated with the test description you are creating; it is not the remark logged with the *COPY TEST\_DESCRIPTION* command.

The */REMARK* qualifier replaces the remark currently associated with the test description with the remark you specify. The */NOREMARK* qualifier specifies that no remark be associated with the new test description.

## ***/TEMPLATE=file-specification***

## ***/NOTEMPLATE***

If you do not specify this qualifier, the existing template file is copied for the new test. If you specify the */TEMPLATE* qualifier, the existing template file is replaced by the specified template file. If you specify the */NOTEMPLATE* qualifier, DEC/Test Manager creates the template file name in the form *test-name.COM* for a noninteractive test and *test-name.SESSION* for an interactive or DECwindows test.

# COPY TEST\_DESCRIPTION

## ***/NOVARIABLES***

Specifies that no variables be associated with the new test description. If you do not specify this qualifier, the variables associated with the old test description are associated with the new test description.

---

## **Example**

```
DTM> COPY TEST_DESCRIPTION MAIL_TEST SHOW_ALL_TEST/PROLOGUE=NEWPRO.COM
_Remark: SHOW ALL test with new prologue file
%DTM-I-DEFAULTED, benchmark file name defaulted to SHOW_ALL_TEST.BMK
%DTM-S-COPIED, test description MAIL_TEST copied
-DTM-S-CREATED, test description SHOW_ALL_TEST created
DTM>
```

This example creates a copy of the existing test description, **MAIL\_TEST**, and names the copy, **SHOW\_ALL\_TEST**. The prologue file named **NEWPRO.COM** is associated with the new test description.

# CREATE COLLECTION

---

## CREATE COLLECTION

Designates a set of tests as a collection.

---

### Format

**CREATE COLLECTION** *collection-name test-group-expression*  
*[/qualifier...] "remark"*

#### Command Qualifiers

**/[NO]BENCHMARK\_DIRECTORY=**directory-specification  
**/CLASS=**(keyword=class-name,...)  
**/[NO]COMPARE[=**(keyword,...]  
**/[NO]EPILOGUE=**file-specification  
**/[NO]LOG**  
**/[NO]PROLOGUE=**file-specification  
**/SENTINEL=**("begin-delimiter","end-delimiter")  
**/[NO]SUBMIT[=**keyword,...]  
**/[NO]TEMPLATE\_DIRECTORY=**directory-specification  
**/VARIABLE=**(variable-name=variable-value,...)  
**/[NO]VERIFY**

#### Defaults

See text  
See text  
**/COMPARE**  
Current collection epilogue  
**/LOG**  
Current collection prologue  
None  
**/NOSUBMIT**  
See text  
See text  
**/VERIFY**

#### Parameter Qualifiers

**/GROUP**  
**/TEST\_DESCRIPTION**

#### Defaults

**/TEST\_DESCRIPTION**  
**/TEST\_DESCRIPTION**

---

### Command Parameters

#### ***collection-name***

Identifies a set of tests that are run as a collection. A collection name consists of up to 39 characters. You cannot use wildcards to specify the collection name parameter. The collection name cannot begin with DTM\$; names with this prefix are reserved for use by DEC/Test Manager.

# CREATE COLLECTION

## ***test-group-expression***

Specifies items of a test expression or a group expression, including test names, group names, and wildcard forms of these names. Separate items in a test group expression with commas. Identify each item as either a test description or a group with the /GROUP or /TEST\_DESCRIPTION parameter qualifiers.

## ***remark***

Specifies a string that contains a comment. You must specify a remark within quotation marks ( " " ); the exception to this rule is when you specify a remark string at the remark prompt.

If you do not provide a remark string, you are prompted for one. However, a null remark is permitted.

---

## **Description**

The CREATE COLLECTION command organizes a set of files that will be treated as a single entity for running tests. A collection can contain any combination of test types. You can execute a collection of tests either interactively or in batch mode. However, DECwindows tests require a connection to a workstation's DECwindows server to run in batch mode.

The CREATE COLLECTION command constructs a set of tests by taking a "snapshot" of the test descriptions at the time the collection is created. Therefore, any changes subsequently made to test descriptions contained in the collection are not reflected in the collection. However, changes made to files referenced by the collection may affect the collection at run time.

You can execute a collection interactively by using the RUN command, or noninteractively in batch mode by using the /SUBMIT command qualifier. If you choose to execute your tests in batch mode, DEC/Test Manager uses the collection name you specified as the name of the batch job. You can also execute a collection again with either the RUN or SUBMIT command.

DEC/Test Manager verifies the existence of files associated with the test descriptions in the collection when the collection is created. If a test description you include in a collection does not exist, the collection is not created. The collection is also not created if a variable you specify (on the CREATE COLLECTION command line) is not global or does not exist.

# CREATE COLLECTION

If you specify the `/NOVERIFY` qualifier, DEC/Test Manager creates the collection without verifying the existence of files associated with tests in the collection. When the collection is executed, DEC/Test Manager may not execute the tests if files that are supposed to be associated with them are missing.

---

## Command Qualifiers

**`/BENCHMARK_DIRECTORY=directory-specification`**

**`/NOBENCHMARK_DIRECTORY`**

Determines whether DEC/Test Manager should search the default benchmark directory for benchmark files for the specified collection.

If you do not include a directory, DEC/Test Manager searches the default benchmark directory for the benchmark file established by the `SET BENCHMARK_DIRECTORY` command.

If you include a directory in the benchmark file specification for a test within the collection, DEC/Test Manager searches that directory for the benchmark file and (if found) overrides the default directory. The directory you specify can be either another directory or a CMS library.

The `/NOBENCHMARK_DIRECTORY` qualifier overrides the default benchmark directory for the specified collection. DEC/Test Manager searches your default directory for all benchmark files without directory specifications.

**`/CLASS=(keyword=class-name,...)`**

Specifies the optional CMS class for benchmark files and template files stored in CMS libraries. The keywords, `BENCHMARK` and `TEMPLATE`, designate the name of the specific set of generations of elements. If you do not specify a class and the file is stored in a CMS library, the latest generation on the main line of descent is used. See the *Guide to VAX DEC/Code Management System* for more information about classes.

You can specify the same class names for your benchmark and template files. If you specify both keywords, separate them with a comma and enclose the list in parentheses. If you specify only one keyword, omit the parentheses.

# CREATE COLLECTION

***/COMPARE[=(keyword,...)]***

***/NOCOMPARE***

Determines whether DEC/Test Manager compares the results of each test with its benchmark file (the file that contains expected test results) after the collection is executed. The default is */COMPARE*.

The */COMPARE* qualifier specifies that DEC/Test Manager is to compare all tests after the collection is executed. A collection must be compared before it can be reviewed. Any differences between the results for a test and its benchmark file are recorded in a difference file for that test. Tests without benchmarks can be compared, but will be marked with the comparison status of new test.

When you review tests, you can have benchmark files generated for them. When you enter the */COMPARE* qualifier, the *COMPARE* command default qualifiers (*/SCREENS*, */LOG*, and */WIDTH=132*) are in effect. You can optionally specify any of the following *COMPARE* command qualifiers as keywords:

**CHARACTERS**  
**FULL**  
**IGNORE=keyword**  
**[NO]PARALLEL**  
**RECORDS**  
**SCREENS**  
**WIDTH**

The */COMPARE* qualifier keywords have the same effect as the *COMPARE* command qualifiers. See the *COMPARE* command qualifiers for a description of the */COMPARE* qualifier keywords.

If you specify more than one keyword, separate the keywords with commas and enclose the list in parentheses. If you specify only one keyword, you can omit the parentheses.

The */NOCOMPARE* qualifier prevents the automatic comparison that DEC/Test Manager ordinarily performs when the collection is executed. You can use the *COMPARE* command later to compare test results for collections created with the */NOCOMPARE* qualifier.

# CREATE COLLECTION

***/EPILOGUE=file-specification***

***/NOEPILOGUE***

Determines whether the default collection epilogue is run with this collection.

The */EPILOGUE* qualifier overrides the default collection epilogue file for this collection. The */NOEPILOGUE* qualifier runs a collection without a collection epilogue. This qualifier has no effect on individual test epilogues.

***/LOG (D)***

***/NOLOG***

Controls whether DEC/Test Manager displays informational and success messages on your screen.

***/PROLOGUE=file-specification***

***/NOPROLOGUE***

Determines whether the default collection prologue is run with this collection.

The */PROLOGUE* qualifier overrides the default collection prologue file for this collection. The */NOPROLOGUE* qualifier runs a collection without a collection prologue file. This qualifier has no effect on individual test prologues.

***/SENTINEL=("begin-delimiter", "end-delimiter")***

Specifies a pair of strings used to delimit a section of text to be ignored during the comparison of result and benchmark files for a noninteractive test. The delimiters can be up to 256 characters per line, and must be unique. Any text between and including the delimiters is treated as if it did not exist.

If you do not enclose the sentinel strings in quotation marks, they are converted to uppercase before the comparison of the files. Sentinel strings may contain any characters, but if you include spaces or tabs, they must be enclosed in quotation marks.

***/SUBMIT[=(keyword,...)]***

***/NOSUBMIT (D)***

Determines whether the collection is executed immediately after it is created. You can submit collections that contain DECwindows tests but DEC/Test Manager must be connected to a DECwindows server for the tests to execute.



# CREATE COLLECTION

The /SUBMIT qualifier executes the collection in batch mode immediately after the collection is created. When you enter the /SUBMIT qualifier, the SUBMIT command qualifiers (/KEEP and /LOG) are in effect. You can optionally specify any of the following SUBMIT command qualifiers as keywords:

AFTER	[NO]CHARACTERISTICS	CPUTIME
[NO]HOLD	[NO]KEEP	[NO]LOG_FILE
NAME	[NO]NOTIFY	[NO]PRINTER
PRIORITY	QUEUE	[NO]USER
WSDEFAULT	WSEXTENT	WSQUOTA

If you specify more than one keyword, separate the keywords with commas and enclose the list in parentheses. If you specify only one keyword, you can omit the parentheses.

The /NOSUBMIT qualifier creates the collection without submitting it to the batch queue. To run the collection, use the SUBMIT command.

***/TEMPLATE\_DIRECTORY=directory-specification***

***/NOTEMPLATE\_DIRECTORY***

Determines whether DEC/Test Manager should search the default template directory for template files for the specified collection.

If you do not include a directory, DEC/Test Manager searches the default template directory for the template file established by the SET TEMPLATE\_DIRECTORY command.

If you include a directory in the template file specification for a test within the collection, DEC/Test Manager searches that directory for the template file. The directory you specify can be either another directory or a CMS library.

The /NOTEMPLATE\_DIRECTORY qualifier overrides the default template directory for the specified collection. DEC/Test Manager searches your default directory for all template files without directory specifications.

***/VARIABLE=(variable-name=variable-value,...)***

Overrides the values of the specified global variables for this collection. If you override the value for more than one variable, separate the variables with commas and enclose the list in parentheses. If you override only one variable, omit the parentheses.

# CREATE COLLECTION

## ***/VERIFY (D)***

## ***/NOVERIFY***

Specifies whether DEC/Test Manager is to verify the existence of files associated with all test descriptions before creating the collection. If a referenced file does not exist, DEC/Test Manager does not create the collection.

The */NOVERIFY* qualifier causes DEC/Test Manager to create the collection without verifying the existence of files associated with all test descriptions before creating the collection. If a file associated with a test description is missing when a collection executes, DEC/Test Manager may not run that test.

---

## Parameter Qualifiers

### ***/GROUP***

Identifies the immediately preceding item in the test group expression as a group. If a test group expression is a list, use this qualifier after each item in the list that designates a group. The default is */TEST\_DESCRIPTION*.

### ***/TEST\_DESCRIPTION***

Identifies the immediately preceding item in the test group expression as a test expression. This is the default.

---

## Examples

1. DTM> CREATE COLLECTION MAIL\_COLL MAIL\*/NOPROLOGUE  
\_Remark: Tests of MAIL commands  
%DTM-S-CREATED, collection MAIL\_COLL created

This example creates the collection MAIL\_COLL. It uses a qualifier to specify that there is to be no collection prologue file associated with this collection, and uses wildcards to specify which tests go into the collection.

## CREATE COLLECTION

- DTM> CREATE COLLECTION MAIL\_PLUS MAIL\*, -  
\_DTM> SEND\_NONINT/GROUP  
\_Remark: More MAIL tests  
%DTM-S-CREATED, collection MAIL\_PLUS created

**This example creates the collection MAIL\_PLUS. The test group expression specifies all tests that begin with MAIL, and all tests in the group SEND\_NONINT at the time the collection is created.**

# CREATE GROUP

---

## CREATE GROUP

Creates a group in the DEC/Test Manager library.

---

### Format

**CREATE GROUP** *group-name* [/qualifier] "remark"

Command Qualifier	Default
[/NO]LOG	/LOG

---

### Command Qualifier

**/LOG (D)**

**/NOLOG**

Controls whether DEC/Test Manager displays informational and success messages on your screen.

---

### Command Parameters

#### ***group-name***

Identifies a group—a category you create to organize tests. A group name consists of up to 39 characters and follows the same syntax rules as for file names.

You cannot use wildcards to specify the group name parameter. You cannot begin the group name with DTM\$; names with this prefix are reserved for use by DEC/Test Manager. A group name must be unique among group names in this DEC/Test Manager library. DEC/Test Manager informs you of any error in naming.

#### ***remark***

Specifies a string that contains a comment. You must specify a remark within quotation marks ( " "). The exception to this rule is when you specify a remark string at the remark prompt. If you do not provide a remark string, you are prompted for one. However, a null remark is permitted.

---

## Description

The **CREATE GROUP** command creates an empty group in the DEC/Test Manager library. After you create a group, you can include test descriptions and other groups in the group with the **INSERT TEST\_DESCRIPTION** and the **INSERT GROUP** commands.

---

## Example

```
DTM> CREATE GROUP SEND_NONINT
_Remark: Tests of MAIL commands that send text (SEND, REPLY, etc)
%DTM-S-CREATED, group SEND_NONINT created
```

This example creates the group **SEND\_NONINT**.

# CREATE LIBRARY

---

## CREATE LIBRARY

Creates a DEC/Test Manager library in an empty VMS directory.

---

### Format

**CREATE LIBRARY** *directory-specification* [/qualifier] "remark"

Command Qualifier	Default
/[NO]LOG	/LOG

---

### Restrictions

- Do not create subdirectories of the directory containing the DEC/Test Manager library. DEC/Test Manager recognizes that they are not part of the library and may delete them.
- Do not create or modify files in the DEC/Test Manager library and do not delete files from the DEC/Test Manager library.
- Do not access the DEC/Test Manager library with commands other than DEC/Test Manager commands. Use only DEC/Test Manager Review subsystem commands to access test run output files.

---

### Command Parameters

#### *directory-specification*

Specifies an empty directory that you have created with the DCL CREATE/DIRECTORY command. The directory specification must follow VMS specifications for directory names. Do not specify your current default directory or a directory that contains files.

#### *remark*

Specifies a string that contains a comment. You must specify a remark within quotation marks ( " " ). The exception to this rule is when you specify a remark string at the remark prompt. If you do not provide a remark string, you are prompted for one. However, a null remark is permitted.

---

## Description

The **CREATE LIBRARY** command creates a DEC/Test Manager library in an empty VMS directory. The library contains the files that DEC/Test Manager needs to describe, run, and review tests.

You can store benchmark files and template files in the DEC/Test Manager library or in CMS libraries. You must store all other files outside the DEC/Test Manager library in another directory or in a CMS library. If you do this, you must inform DEC/Test Manager where the files are stored.

You may find it useful to create a separate library for each project for which you use DEC/Test Manager.

---

## Command Qualifier

**/LOG (D)**

**/NOLOG**

Controls whether DEC/Test Manager displays informational and success messages on your screen.

---

## Example

```
$ CREATE/DIR [USER01.DTMLIB]
$ DTM
DTM>
DTM> CREATE LIBRARY [USER01.DTMLIB]
%DTM-S-CREATED, DTM library DUA0:[USER01.DTMLIB] created
```

This example shows how to create a DEC/Test Manager library by first creating an empty directory, **DTMLIB.DIR**, and then using the **CREATE LIBRARY** command to turn this directory into a DEC/Test Manager library.

# CREATE TEST\_DESCRIPTION

---

## CREATE TEST\_DESCRIPTION

Creates a test description in the DEC/Test Manager library.

---

### Format

**CREATE TEST\_DESCRIPTION** *test-name* [/qualifier...] "remark"

#### Command Qualifiers

/BENCHMARK=file-specification  
/COMMAND="DCL-command"  
/COMPARISON\_TYPE=keyword  
/DECWINDOWS  
/EPILOGUE=file-specification  
/FILTER=(keyword,...)  
/[NO]INTERACTIVE  
/[NO]LOG  
/PROLOGUE=file-specification  
/TEMPLATE=file-specification  
/VARIABLE=(variable-name[=variable-value],...)

#### Defaults

/BENCHMARK=test-name.BMK  
None  
None  
None  
None  
None  
/NOINTERACTIVE  
/LOG  
None  
See text  
None

---

### Restrictions

- The /COMMAND qualifier applies to DECwindows tests only.
- The /FILTER qualifier applies to noninteractive and terminal tests only.

---

### Command Parameters

#### *test-name*

Specifies a unique name for a test description. A test name consists of up to 39 characters and follows the same syntax rules as for file names. You cannot use wildcards to specify the test name parameter. The test name cannot begin with DTM\$ because names with this prefix are reserved for use by DEC/Test Manager.



# CREATE TEST\_DESCRIPTION

## *remark*

Specifies a string that contains a comment. You must specify a remark within quotation marks (" "). The exception to this rule is when you specify a remark string at the remark prompt. If you do not provide a remark string, you are prompted for one. However, a null remark is permitted.

---

## Description

The CREATE TEST\_DESCRIPTION command creates a test description in the DEC/Test Manager library. A test description is all the information associated with a specific test.

The test name parameter specifies a value for the test name field of the test description. You can specify values for the other test description fields by supplying values for the appropriate command qualifiers.

The test name is the unique identifier of the test description and is the only means of access to a test description. The /TEMPLATE qualifier specifies the template file for this test. A template file is a command file that is a test or a recorded interactive session. If you do not specify a template file name at the time you create a test description, DEC/Test Manager supplies a default file name of the form test-name.COM unless you specified the /INTERACTIVE or /DECWINDOWS qualifier, in which case the default file name is test-name.SESSION.

When you create a test description, you can include file specifications for the template, benchmark, prologue, or epilogue field, regardless of whether the specified file exists. DEC/Test Manager does not check whether the files exist; it creates the test description with the file names you supply or with default file names.

If you specify a name for the variable field, the variable must exist. If the variable does not exist, DEC/Test Manager will not create the test description.

When you create a collection to run your test, DEC/Test Manager searches for all the files specified in the test description. If a file is missing, the collection is not created. If you specify the /NOVERIFY qualifier on the CREATE COLLECTION command, DEC/Test Manager creates the collection without verifying that the files exist.

# CREATE TEST\_DESCRIPTION

When the collection is created, DEC/Test Manager resolves the logical name. If you use a logical name in a file specification for a CREATE TEST\_DESCRIPTION qualifier value, the logical name is not resolved until you place the test in a collection. If you specify the /NOVERIFY qualifier on the CREATE COLLECTION command, DEC/Test Manager creates the collection without resolving the logical name.

---

## Command Qualifiers

### ***/BENCHMARK=file-specification***

Specifies the file to contain the expected output from the test's execution. DEC/Test Manager supplies a file name of the form test-name.BMK. If your file specification includes a directory specification, it overrides the default benchmark directory for the library. Benchmark files may be located in the DEC/Test Manager library, in another directory, or in a CMS library.

### ***/COMMAND="DCL-command"***

Specifies a command to be executed before the test is recorded or executed. Use this qualifier to start applications for inclusion in the test. This qualifier applies to DECwindows tests only.

### ***/COMPARISON\_TYPE=keyword***

Specifies how the result and benchmark files are to be compared. A comparison type is not associated with the test description. The valid values for keyword are as follows:

---

<b>Keyword</b>	<b>Meaning</b>
CHARACTERS	Compares files character by character.
RECORDS	Compares files record by record. This is the default for noninteractive terminal tests.
SCREENS	Compares files screen by screen; screens not marked are not compared. This is the default for interactive terminal and DECwindows tests.

---

DECwindows tests can only use the SCREENS comparison type. The SCREENS comparison type is also the default comparison type for interactive terminal tests. If you specify the /COMPARISON\_TYPE=SCREENS qualifier for a noninteractive test, this value is ignored.

# CREATE TEST\_DESCRIPTION

## ***/DECWINDOWS***

Specifies that the test being created is marked as a DECwindows test.

## ***/EPILOGUE=file-specification***

Adds the specified epilogue file to the test description. The test epilogue file is run immediately after the test template file is executed. This epilogue file is unrelated to the collection epilogue file.

You cannot store epilogue files in the DEC/Test Manager library; store them in another directory or in a CMS library.

## ***/FILTER=(keyword,...)***

Available for interactive and noninteractive terminal tests only, the **/FILTER** qualifier selects one or more filters to remove run-time data from the result file that the test run produces. The valid values for keyword are as follows:

<b>Keyword</b>	<b>Filter</b>
ALL	Specifies that all the filters in this table be used
DATE	<p>Where the date form is abbreviated, the date filter replaces date stamps by substituting a “d” for each displayed number of the day of the month, an “m” for each displayed letter of the month, and a “y” for each displayed number of the year. Where the date form is spelled out, the month name is replaced by “month”, the numeric day is replaced by “day”, and the year is replaced by “year”.</p> <p>The following list shows some examples of the date filtering functions; this list is not all inclusive.</p> <ul style="list-style-type: none"><li>17-OCT-1989 with dd-mmm-yyyy</li><li>17 OCT 89 with dd mmm yy</li><li>89.OCT.17 with yy.mmm.dd</li><li>10/17/89 with mm/dd/yy</li><li>1989/10/17 with yyyy/mm/dd</li><li>October 17, 1989 with month day, year</li><li>Oct. 17, 1989 with month day, year</li><li>17.October.1989 with day.month.year</li><li>89-October-17 with year-month-day</li></ul>

# CREATE TEST\_DESCRIPTION

Keyword	Filter
TIME	Replaces time stamps with the following forms:  15:37:53.22 with hh:mm:ss.xxxx 15:37:53 with hh:mm:ss 15:37 with hh:mm 3:37 PM with hh:mm xm 15H37m with hhHmmm 15H37' with hhHmm' 15.37 h with hh.mm h 15 h 37"53 s with hh h mm"ss s 15 h 37 min with hh h mm min kl 15.37 with kl hh.mm h 15.37 with h hh.mm
FILE_NAMES	Replaces the file names with FILENAME.EXT
DIRECTORIES	Replaces the directory specification field in the file specification with DISK:[DIRECTORY]
TRACE_BACK	Replaces 8-bit memory addresses with xxxxxxxx
VERSION	Replaces file versions with VERSION

If you specify more than one keyword, separate the keywords with commas and enclose the list in parentheses. If you specify only one keyword, omit the parentheses.

## ***/INTERACTIVE***

### ***/NOINTERACTIVE (D)***

Specifies whether the test being created is marked as an interactive terminal test.

The ***/INTERACTIVE*** qualifier marks a test description as containing an interactive terminal test. The ***/NOINTERACTIVE*** qualifier marks a test description as containing a noninteractive test.

## ***/LOG (D)***

### ***/NOLOG***

Controls whether DEC/Test Manager displays informational and success messages on your screen.

## ***/PROLOGUE=file-specification***

Adds the specified prologue file to the test description.

# CREATE TEST\_DESCRIPTION

The test prologue file is run immediately before the test template file is executed. This prologue file is unrelated to the collection prologue file.

You cannot store prologue files in the DEC/Test Manager library; store them in another directory or in a CMS library.

## ***/TEMPLATE=file-specification***

Specifies the command file that runs a test, the file containing an interactive terminal or DECwindows session. DEC/Test Manager supplies a template file name of the form test-name.COM for noninteractive tests and test-name.SESSION for interactive and DECwindows tests. If your file specification includes a directory specification, DEC/Test Manager ignores the default template directory.

You cannot store template files that you create, except SESSION files, in the DEC/Test Manager library; store them in another directory, or in a CMS library.

## ***/VARIABLE=(variable-name[=variable-value],...)***

Enables you to associate existing variables with the test description you are creating. The variables you specify must be defined in the DEC/Test Manager library by using the CREATE VARIABLE command. A variable associated with a test description by this qualifier is local in scope.

The /VARIABLE qualifier also enables you to redefine values for the variables you specify. If you specify an optional value, the variable takes on that value only for this test description; the value of the original variable is unaffected.

If you specify more than one variable name, separate the names with commas and enclose the list in parentheses. If you specify only one variable name, omit the parentheses. You cannot use wildcards.

---

## Examples

1. DTM> CREATE TEST\_DESCRIPTION SEND\_MAIL\_TEST  
\_Remark: Send a message test.  
%DTM-I-DEFAULTED, benchmark file name defaulted to SEND\_MAIL\_TEST.BMK  
%DTM-I-DEFAULTED, template file name defaulted to SEND\_MAIL\_TEST.COM  
%DTM-S-CREATED, test description SEND\_MAIL\_TEST created.

This example creates a noninteractive test description with the test name SEND\_MAIL\_TEST.

## CREATE TEST\_DESCRIPTION

```
2. DTM> CREATE TEST_DESCRIPTION/TEMPLATE=MAIL_INT.COM -
   _DTM> /INTERACTIVE/PROLOGUE=NOBROADCAST.COM/EPILOGUE=BROADCAST.COM
   _test name: MAIL_TEST_INT
   _Remark: Creating a MAIL test
   %DTM-I-DEFAULTED, benchmark file name defaulted to MAIL_TEST_INT.BMK
   %DTM-I-DEFAULTED, template file name defaulted to MAIL_TEST_INT.SESSIO
   %DTM-S-CREATED, test description MAIL_TEST_INT created
```

**This example creates an interactive test description MAIL\_TEST\_INT and includes template, prologue, and epilogue file names. Note that DEC/Test Manager prompts for the test name and remark.**

---

# CREATE VARIABLE

Defines a variable in the DEC/Test Manager library.

---

## Format

**CREATE VARIABLE** *variable-name variable-value [/qualifier...]*  
*"remark"*

<b>Command Qualifiers</b>	<b>Defaults</b>
/GLOBAL	/LOCAL
/LOCAL	/LOCAL
/[NO]LOG	/LOG
/LOGICAL	/SYMBOL
/NUMERIC	See text
/STRING	See text
/SYMBOL	/SYMBOL

---

## Command Parameters

### ***variable-name***

Specifies a variable's name, which must be unique, consist of up to 39 characters, and follow VMS rules for file names. You cannot use wildcards, the variable names P1 through P8, or variable names beginning with DTM\$, which are reserved for use by DEC/Test Manager.

### ***variable-value***

Specifies the variable's value. This value remains in effect until you redefine it for a particular test description.

### ***remark***

Specifies a string that contains a comment. You must specify a remark within quotation marks (" "). The exception to this rule is when you specify a remark string at the remark prompt. If you do not provide a remark string, you are prompted for one. However, a null remark is permitted.

# CREATE VARIABLE

---

## Description

The **CREATE VARIABLE** command defines a variable in the current DEC/Test Manager library. Only variables defined in the library can be included in test descriptions or collections.

You specify the default value for a variable with the variable value parameter. If you do not supply a variable value, DEC/Test Manager prompts you for one. Optional qualifiers enable you to specify how the variable is to be handled during processing. You can override this value for a particular test description.

You specify the value of a variable with the variable value parameter. You specify its use as a local or global variable with the **/GLOBAL** or **/LOCAL** qualifier. You specify its use as a symbol or logical name with the **/SYMBOL** or **/LOGICAL** qualifier.

If you do not specify the **/SYMBOL** or **/LOGICAL** qualifier, DEC/Test Manager creates the variable as a symbol. If a variable is defined as a symbol, you must further define the variable value as either a text string by using the **/STRING** qualifier, or a numeric value by using the **/NUMERIC** qualifier. If you do not use one of these qualifiers, DEC/Test Manager interprets a quoted variable value as a text string and an unquoted variable value as a numerical value.

When using the **/NUMERIC** and **/STRING** qualifiers, you must place DCL operators such as the plus sign (+) and minus (-) in a quoted string to be used as arithmetic operators. DEC/Test Manager attempts to prevent the generation of syntactically incorrect DCL assignment statements. Therefore, numeric symbol variables and logical variables cannot have a null value.

---

## Command Qualifiers

### ***/GLOBAL***

Defines the variable as being accessible to all tests in all collections. You cannot specify both **/LOCAL** and **/GLOBAL** with the same **CREATE VARIABLE** command. The default is **/LOCAL**.



# CREATE VARIABLE

## ***/LOCAL***

Defines the variable as being accessible only to an individual test that references it in its test description. The default is ***/LOCAL***. You cannot specify both ***/LOCAL*** and ***/GLOBAL*** with the same **CREATE VARIABLE** command.

## ***/LOG (D)***

## ***/NOLOG***

Controls whether DEC/Test Manager displays informational and success messages on your screen.

## ***/LOGICAL***

Defines the variable as a VMS logical name. You cannot specify both ***/LOGICAL*** and ***/SYMBOL*** with the same **CREATE VARIABLE** command. The default is ***/SYMBOL***.

## ***/NUMERIC***

Used only with the ***/SYMBOL*** qualifier, the ***/NUMERIC*** qualifier defines the symbol type as a numeric value. Use this qualifier to define a quoted symbol value as numeric. You cannot specify both ***/NUMERIC*** and ***/STRING*** with the same **CREATE VARIABLE** command. If the variable value is not enclosed in quotation marks ( " "), the variable type is defined as a numeric value.

## ***/STRING***

Used with the ***/SYMBOL*** qualifier, the ***/STRING*** qualifier defines a symbol type as a text string. Use this qualifier to define an unquoted symbol value as a text string. You cannot specify both ***/NUMERIC*** and ***/STRING*** with the same **CREATE VARIABLE** command. If the variable value is enclosed in quotation marks ( " "), the variable type is defined as a text string.

## ***/SYMBOL***

Defines the variable to be a VMS symbol. When you specify a variable as a symbol with the ***/SYMBOL*** qualifier, you must further define it as either a numeric value (with the ***/NUMERIC*** qualifier) or a text string (with the ***/STRING*** qualifier). You cannot specify both ***/LOGICAL*** and ***/SYMBOL*** with the same **CREATE VARIABLE** command. The default is ***/SYMBOL***.

# CREATE VARIABLE

---

## Example

```
DTM> CREATE VARIABLE/SYMBOL/LOCAL INPUT_FILE "emptyfil"  
_Remark: Name of input file, with an empty file as the default  
%DTM-S-CREATED, symbol variable INPUT_FILE created.
```

This example creates the variable `INPUT_FILE`. It is defined as a local symbol, with a default value of `emptyfil`. The quotation marks indicate that the value is a string.

---

## DEFINE/KEY

Defines a key to execute a command string.

---

### Format

**DEFINE/KEY** *key-name* "*command-string*" [*/qualifier...*]

#### Command Qualifiers

/[NO]ECHO  
 /[NO]IF\_STATE=(state-name,...)  
 /[NO]LOCK\_STATE  
 /[NO]SET\_STATE=state-name  
 /[NO]TERMINATE

#### Defaults

/ECHO  
 Current state  
 /NOLOCK\_STATE  
 Current state  
 /NOTERMINATE

---

### Restrictions

- Key definitions apply only to terminal environments.
- The DEFINE/KEY command can only be used at the DEC/Test Manager subsystem level.

---

### Command Parameters

#### ***key-name***

Specifies the key to define. You can use the DEFINE/KEY command to define the following keys:

PF1 to PF4  
 KP0 to KP9  
 period (.)  
 comma (,)  
 minus (-)  
 Enter  
 left arrow (←)  
 right arrow (→)

# DEFINE/KEY

Find (E1)  
Insert Here (E2)  
Remove (E3)  
Select (E4)  
Prev Screen (E5)  
Next Screen (E6)  
Help  
Do  
F6 to F20

## ***command-string***

Specifies the command string to be entered when you press the defined key. The command string can be a DEC/Test Manager command. If the command string contains any spaces, enclose the command string in quotation marks (" ").

---

## **Description**

The DEFINE/KEY command defines a key to issue a DEC/Test Manager command. You can customize your keyboard by defining keys to issue often used commands or command strings that are long.

The definitions you create with the DEFINE/KEY command are in effect only for the current DEC/Test Manager session; the next time you invoke DEC/Test Manager, only the default key definitions will be in effect. To save your key definitions for use in every DEC/Test Manager session, include them in a DEC/Test Manager initialization file. This file is executed whenever you invoke DEC/Test Manager as a subsystem. See Section 6.6.2 for more information on the initialization file.

If you have key definitions that you want to save but do not necessarily want to use every time you invoke DEC/Test Manager, define them in a command procedure.

DEC/Test Manager provides a set of default definitions. You can use the DEFINE/KEY command to replace these definitions or to define certain undefined keys. Pressing the PF2 key displays the default key definitions.

The state name value used with the /IF\_STATE, /LOCK\_STATE, and /SET\_STATE qualifiers can be any alphanumeric string. The state names defined by DEC/Test Manager are DTM and GOLD\_DTM.

You can define the GOLD key (PF1) to execute DEC/Test Manager commands in two keystrokes by using the DEFINE/KEY command with the /SET\_STATE=GOLD\_DTM qualifier. By doing this, you can provide two definitions to the same key. For example, you can define KP1 to issue the CREATE GROUP command and define GOLD-KP1 to issue the MODIFY GROUP command.

---

## Command Qualifiers

### ***/ECHO (D)***

### ***/NOECHO***

Specifies whether the command is displayed on your screen after you press the defined key. You cannot specify both the /NOECHO qualifier and the /NOTERMINATE qualifier.

### ***/IF\_STATE=(state-name,...)***

### ***/NOIF\_STATE***

Specifies a list of states, any one of which must be set to enable the specified key definition. The default is the current state. DEC/Test Manager defines the two state names as DTM and GOLD\_DTM. The /NOIF\_STATE qualifier selects the current state.

### ***/LOCK\_STATE***

### ***/NOLOCK\_STATE (D)***

Specifies the state specified with the /SET\_STATE qualifier until you use the /SET\_STATE qualifier again to change it.

### ***/SET\_STATE=state-name***

### ***/NOSET\_STATE***

Associates a state with the key you are defining. The default is the current state. DEC/Test Manager defines the two state names as DTM and GOLD\_DTM. You cannot define a key specifying both the /SET\_STATE qualifier and the /TERMINATE qualifier. The /NOSET\_STATE qualifier selects the current state.

# DEFINE/KEY

## ***/TERMINATE*** ***/NOTERMINATE (D)***

Determines whether the specified command string executes when you press the defined key. When you use the */NOTERMINATE* qualifier, you must press the RETURN key to execute a command. You cannot specify both the */SET\_STATE* qualifier and the */TERMINATE* qualifier or the */NOECHO* qualifier with the */NOTERMINATE* qualifier.

---

## Examples

1. DTM> DEFINE/KEY KP5 "SET LIBRARY DUA0:[USER01.LIB\_A]"/TERMINATE  
DTM>

If you subsequently press keypad 5, the following text is displayed:

```
DTM> SET LIBRARY DUA0:[USER01.LIB_A]
%DTM-S-LIBIS, DEC/Test Manager library is DUA0:[USER01.LIB_A]
DTM>
```

2. DTM> DEFINE/KEY KP5 /IF\_STATE=GOLD\_DTM -  
\_DTM> "SET LIBRARY DUA0:[USER01.LIB\_B]"/TERMINATE  
DTM>

This example defines GOLD keypad 5 to set the default DEC/Test Manager library to a different library than the one in the previous example. If you subsequently press GOLD Keypad 5, the following text is displayed:

```
DTM> SET LIBRARY DUA0:[USER01.LIB_B]
%DTM-S-LIBIS, DEC/Test Manager library is DUA0:[USER01.LIB_B]
DTM>
```

---

## DELETE COLLECTION

Deletes the specified collection and any associated difference and result files from the DEC/Test Manager library.

---

### Format

**DELETE COLLECTION** *collection-expression* [/qualifier...]  
"remark"

#### Command Qualifiers

/[NO]CONFIRM

/[NO]LOG

#### Defaults

/CONFIRM

/LOG

---

### Command Parameters

#### *collection-expression*

Specifies the collections to delete. The collection expression can be a collection name or a list of names separated by commas. You can use wildcards.

#### *remark*

Specifies a string that contains a comment. You must specify a remark within quotation marks (" "). The exception to this rule is when you specify a remark string at the remark prompt. If you do not provide a remark string, you are prompted for one. However, a null remark is permitted.

---

### Description

The **DELETE COLLECTION** command deletes one or more specified collections and any related difference or result files from the DEC/Test Manager library.

# DELETE COLLECTION

You cannot delete a collection that is in use. Also, you must have sufficient privileges to delete the files. If the collection run ended abnormally and you are unable to delete a collection, use the **VERIFY/RECOVER** command and then reissue the **DELETE COLLECTION** command.

This command does not affect benchmark files.

---

## Command Qualifiers

***/CONFIRM (D)***

***/NOCONFIRM***

Controls whether DEC/Test Manager prompts you to confirm each deletion. Valid responses are Yes, No, All, or Quit.

***/LOG (D)***

***/NOLOG***

Controls whether DEC/Test Manager displays informational and success messages on your screen. The default is */LOG*.

---

## Examples

1. DTM> DELETE COLLECTION MAIL\_COLL "No longer needed"  
Confirm deletion of collection MAIL\_COLL [Y/N] (N): y  
%DTM-I-DELETED, collection MAIL\_COLL deleted  
%DTM-S-DELETIONS, 1 deletion completed

This example deletes the collection MAIL\_COLL. The default */CONFIRM* qualifier is in effect.

2. DTM> DELETE COLLECTION \*MAIL\* "Deleting all MAIL collections"  
Confirm deletion of collection MAIL\_COLL [Y/N] (N): Y  
%DTM-I-DELETED, collection MAIL\_COLL deleted  
Confirm deletion of collection DELETE\_MAIL\_COLL [Y/N] (N): Y  
%DTM-I-DELETED, collection DELETE\_MAIL\_COLL deleted  
%DTM-S-DELETIONS, 2 deletions completed

This example deletes all the collections that contain MAIL as part of their name. The default */CONFIRM* qualifier is in effect.



---

# DELETE GROUP

Deletes a group from the DEC/Test Manager library.

---

## Format

**DELETE GROUP** *group-expression* [/qualifier...] "remark"

### Command Qualifiers

/[NO]CONFIRM

/[NO]LOG

### Defaults

/CONFIRM

/LOG

---

## Restriction

- You cannot delete a group if it contains any test descriptions or other groups, or if it is a subgroup of another group. If any test descriptions or groups are still in the group when you issue the DELETE GROUP command, DEC/Test Manager reports that the specified group has not been deleted.

---

## Command Parameters

### *group-expression*

Specifies the groups to delete. The group expression can be a group name or a list of group names separated by commas. You can use wildcards.

### *remark*

Specifies a string that contains a comment. You must specify a remark within quotation marks (" "). The exception to this rule is when you specify a remark string at the remark prompt. If you do not provide a remark string, you are prompted for one. However, a null remark is permitted.

# DELETE GROUP

---

## Description

The **DELETE GROUP** command deletes a group from the DEC/Test Manager library.

The **SHOW GROUP/MEMBER** command lists the groups of which the specified group is a member. The **SHOW GROUP/CONTENTS** command lists the groups and test descriptions contained in this group. Use the **REMOVE TEST\_DESCRIPTION** command to remove test descriptions from the group. Use the **REMOVE GROUP** command to remove subgroups of the group or to remove the group from another group.

---

## Command Qualifiers

***/CONFIRM (D)***

***/NOCONFIRM***

Controls whether DEC/Test Manager prompts you to confirm each deletion. Valid responses are Yes, No, All, or Quit.

***/LOG (D)***

***/NOLOG***

Controls whether DEC/Test Manager displays informational and success messages on your screen.

---

## Examples

1. DTM> DELETE GROUP MAIL\_NONINT/NOCONFIRM "Getting rid of this group"  
%DTM-S-DELETED, group MAIL\_NONINT deleted

This example deletes the group **MAIL\_NONINT**. You are not prompted for confirmation because the **/NOCONFIRM** qualifier is in effect.

## DELETE GROUP

```
2. DTM> DELETE GROUP * "Deleting all groups"
Confirm deletion of group MATH [Y/N] (N): ALL
%DTM-I-DELETED, group MATH deleted
%DTM-I-DELETED, group RELOP deleted
%DTM-I-DELETED, group VARS deleted
%DTM-S-DELETIONS, 3 deletions completed
```

**This example deletes all the groups in the library. The default /CONFIRM qualifier is in effect. By typing ALL, you indicate that all groups can be deleted without further requests for confirmation.**

# DELETE HISTORY

---

## DELETE HISTORY

Deletes history information from the history file.

---

### Format

**DELETE HISTORY** *[/qualifier...]* "remark"

<b>Command Qualifiers</b>	<b>Defaults</b>
/BEFORE=time	See text
/[NO]CONFIRM	/CONFIRM
/[NO]LOG	/LOG
/OUTPUT[=file-specification]	/OUTPUT=HISTORY.OUT

---

### Command Parameter

***remark***

Specifies a string that contains a comment. You must specify a remark within quotation marks (" "). The exception to this rule is when you specify a remark string at the remark prompt. If you do not provide a remark string, you are prompted for one. However, a null remark is permitted.

---

### Description

The **DELETE HISTORY** command removes history information from the history file. You specify the history information that you want removed by using the **/BEFORE** qualifier. That information is then placed in a file named **HISTORY.OUT** in your current directory. After you have removed information from the history file, you cannot replace it.

The **DELETE HISTORY** command is logged in the history file in the usual manner. In addition, it is also logged in the file at the point where history information is being deleted; the log entry takes the following form:

```
date time user-name REMARK "PREVIOUS HISTORY DELETED"
```

---

## Command Qualifiers

### ***/BEFORE=time***

Deletes all history information from the history file dated prior to the specified date. The deleted information is replaced by a single entry stating that history information has been deleted from the history file.

If you omit the */BEFORE* qualifier, the default is to remove information that was logged prior to the time you enter the command. If you include the */BEFORE* qualifier and do not specify a time, the default is *TODAY*.

You can specify the time as an absolute, delta, or combination time value, or as one of the following keywords: *TODAY*, *TOMORROW*, or *YESTERDAY*. DEC/Test Manager interprets *TOMORROW* as the time at which you enter the *DELETE HISTORY* command.

### ***/CONFIRM (D)***

### ***/NOCONFIRM***

Controls whether DEC/Test Manager prompts you to confirm each deletion. Valid responses are Yes, No, All, or Quit.

### ***/LOG (D)***

### ***/NOLOG***

Controls whether DEC/Test Manager displays informational and success messages on your screen.

### ***/OUTPUT[=file-specification]***

Sends output from the *DELETE HISTORY* command to the specified file.

The output is written to a file called *HISTORY.OUT* in your current directory if you do not specify the */OUTPUT* qualifier. If you specify the file name without the file type, the file type defaults to *.LIS*.

# DELETE HISTORY

---

## Example

```
DTM> DELETE HISTORY /BEFORE=08-JAN "Deleting old information"
Confirm DELETE HISTORY/BEFORE=8-Jan-1989 [Y/N] (N): Y
%DTM-S-HISTDEL, 150 history records deleted
DTM>
```

This example deletes all history records in the history file recorded before January 8, 1989. The deleted information is replaced with the following record:

```
* 8-JAN-1989 00:00:00 SMITH REMARK "PREVIOUS HISTORY DELETED"
```

---

## DELETE TEST\_DESCRIPTION

Deletes a test description from the DEC/Test Manager library.

---

### Format

```
DELETE TEST_DESCRIPTION  test-expression [/qualifier...]  
                          "remark"
```

<b>Command Qualifiers</b>	<b>Defaults</b>
---------------------------	-----------------

**/[NO]CONFIRM**

**/CONFIRM**

**/[NO]LOG**

**/LOG**

---

### Restrictions

- You cannot delete a test description if it belongs to a group. Use the **REMOVE TEST\_DESCRIPTION** command to remove a test description from a group. Use the **SHOW TEST\_DESCRIPTION/GROUPS** command to display the groups to which the test description belongs.
- Do not delete a test description that is part of an existing collection. If you delete a test description that is part of a collection, you may see error messages when you issue other DEC/Test Manager commands. For example, if you review a collection from which you have deleted a test description and its associated benchmark file for a noninteractive test, you will see a message indicating that the result description contains errors and you will be unable to examine the benchmark file for this result description. If you delete the test description and benchmark file for an interactive test that uses a screen comparison, you will be unable to examine any of the files associated with the test.

# DELETE TEST\_DESCRIPTION

---

## Command Parameters

### *test-expression*

Specifies the test descriptions to delete. The test expression can be a test name or a list of test names separated by commas. You can use wildcards.

### *remark*

Specifies a string that contains a comment. You must specify a remark within quotation marks (" "). The exception to this rule is when you specify a remark string at the remark prompt. If you do not provide a remark string, you are prompted for one. However, a null remark is permitted.

---

## Description

The DELETE TEST\_DESCRIPTION command deletes the specified test description from the library. It also deletes the test description's benchmark file if it exists in the DEC/Test Manager library. If the benchmark file is outside the DEC/Test Manager library (in another directory or in a CMS library), DEC/Test Manager deletes the test description but does not delete the benchmark file.

Result and difference files generated after a test run are not affected by the DELETE TEST\_DESCRIPTION command. See the DELETE COLLECTION command description for information about deleting these files.

---

## Command Qualifiers

### */CONFIRM (D)*

### */NOCONFIRM*

Controls whether DEC/Test Manager prompts you to confirm each deletion. Valid responses are Yes, No, All, or Quit.

### */LOG (D)*

### */NOLOG*

Controls whether DEC/Test Manager displays informational and success messages on your screen.



# DELETE TEST\_DESCRIPTION

---

## Examples

1. DTM> DELETE TEST\_DESCRIPTION/NOCONFIRM RMTEST "Deleting RMTEST"  
%DTM-S-DELETED, test\_description RMTEST deleted

This example deletes the test description RMTEST. The /NOCONFIRM qualifier is in effect.

2. DTM> DELETE TEST\_DESCRIPTION/NOCONFIRM \*TEST\*  
\_Remark: Deleting all MAIL tests  
%DTM-I-DELETED, test\_description MAIL\_TEST deleted  
%DTM-I-DELETED, test\_description RMTEST deleted  
%DTM-I-DELETED, test\_description SEND\_MAIL\_TEST deleted  
%DTM-I-DELETED, test\_description TEST\_TUBE deleted  
%DTM-S-DELETIONS, 4 test\_descriptions deleted

This example deletes from the current library all the test descriptions that contain TEST as part of their name. The /NOCONFIRM qualifier is in effect.

# DELETE VARIABLE

---

## DELETE VARIABLE

Deletes specified variables from the DEC/Test Manager library.

---

### Format

**DELETE VARIABLE** *variable-expression* [/qualifier...] "remark"

Command Qualifiers	Defaults
/[NO]CONFIRM	/CONFIRM
/[NO]LOG	/LOG

---

### Restriction

- If you attempt to delete several variables and one or more of them are associated with test descriptions, DEC/Test Manager deletes only those variables not associated with a test description.

---

### Command Parameters

***variable-expression***

Specifies the variables to delete. The variable expression can be a variable name or a list of variable names separated by commas. You can use wildcards.

***remark***

Specifies a string that contains a comment. You must specify a remark within quotation marks (" "). The exception to this rule is when you specify a remark string at the remark prompt. If you do not provide a remark string, you are prompted for one. However, a null remark is permitted.

---

## Description

The **DELETE VARIABLE** command deletes one or more specified variables from the DEC/Test Manager library.

This command does not delete any variables currently associated with a test description. If you attempt such a deletion, DEC/Test Manager issues an error message. Use the **SHOW VARIABLE/TEST\_DESCRIPTION** command to list the test descriptions with which a variable is associated. Use the **MODIFY TEST\_DESCRIPTION** command to disassociate a variable from a test description.

Variables are described in Chapter 6.

---

## Command Qualifiers

***/CONFIRM (D)***

***/NOCONFIRM***

Controls whether DEC/Test Manager prompts you to confirm each deletion. Valid responses are Yes, No, All, or Quit.

***/LOG (D)***

***/NOLOG***

Controls whether DEC/Test Manager displays informational and success messages on your screen.

---

## Example

```
DTM> DELETE VARIABLE INPUT_FILE "Deleting variable INPUT_FILE"
Confirm deletion of variable INPUT_FILE [Y/N] (N): y
%DTM-S-DELETED, variable INPUT_FILE deleted.
```

This example deletes the variable **INPUT\_FILE**. The default **/CONFIRM** qualifier is in effect.

# DISPLAY

---

## DISPLAY

Displays the benchmark file for a specified interactive terminal test.

---

### Format

**DISPLAY** *test-name* [/qualifier]

**Command Qualifier**  
/BENCHMARK

**Default**  
/BENCHMARK

---

### Command Parameter

***test-name***

Specifies the name of the test description for an interactive terminal test whose benchmark file is to be displayed. You cannot use wildcards to specify the test name parameter.

---

### Description

The **DISPLAY** command displays the benchmark file as it existed when it was created. For example, if the benchmark file contains dates or times that were recorded when the test was created, those dates or times are displayed rather than the current date or time. Commands in the benchmark file are not executed.

The **DISPLAY** command with the **/BENCHMARK** qualifier searches for the benchmark file in the default benchmark directory, unless this default is overridden by a benchmark directory specified on the test description.

---

### Command Qualifier

***/BENCHMARK***

Specifies that the benchmark file associated with the specified interactive terminal test is to be displayed. The default is to display the benchmark file

---

## Example

```
DTM> DISPLAY/BENCHMARK MEMO_TEST
```

This example displays the banner screen, Screen 0, for interactive display. Follow the directions on the screen to display your benchmark file. See Chapter 5 for more information about displaying benchmark files for interactive terminal tests.

# DTM

---

## DTM

Invokes the DEC/Test Manager. You enter the DTM command at the DCL prompt ( \$ ).

---

### Format

**DTM** *[/qualifier]*

**Command Qualifiers**

**/[NO]INIT**

**/INTERFACE=interface**

**Defaults**

**/INIT**

**/INTERFACE=CHARACTER\_CELL**

---

### Description

Issuing the DTM command on the DCL command line invokes DEC/Test Manager and displays the DTM> prompt. To exit from the DEC/Test Manager system, type the EXIT command or press CTRL/Z.

Issuing the DTM command with the /INTERFACE=DECWINDOWS qualifier invokes DEC/Test Manager and displays the DEC/Test Manager main window. The /INTERFACE=DECWINDOWS qualifier is valid only in a workstation environment. To exit from the DEC/Test Manager DECwindows interface, pull down the File menu and choose Exit.

---

### Command Qualifier

***/INIT (D)***

***/NOINIT***

Specifies whether DEC/Test Manager executes the initialization file (defined by the logical name DTM\$INIT) when invoked. The default is to execute any existing initialization file whenever you invoke DEC/Test Manager.

***/INTERFACE=interface***

Specifies that DEC/Test Manager is to run in the character cell (terminal) or DECwindows environment. The options for this qualifier are CHARACTER\_CELL (the default) and DECWINDOWS.

---

**Example**

```
$ DTM  
DTM>
```

This example shows how to invoke the DEC/Test Manager system.

# EXIT

---

## EXIT

Terminates a DEC/Test Manager session.

---

## Format

### EXIT

**Command Qualifiers**

None

**Defaults**

None

---

## Description

The **EXIT** command terminates a DEC/Test Manager session and returns control to the DCL command line level. You can also press CTRL/Z to terminate a DEC/Test Manager session.

---

## Example

```
DTM> EXIT  
$
```

This example terminates a DEC/Test Manager session.



---

## EXTRACT

Extracts an input file from an interactive terminal or DECwindows session file.

---

### Format

**EXTRACT** *session-file-specification* [*input-file-specification*]  
 [/*qualifier...*]

#### Command Qualifiers

/DECWINDOWS  
 /INTERACTIVE  
 /[NO]LOG  
 /TERMINATION\_CHARACTER=character

#### Defaults

/INTERACTIVE  
 /INTERACTIVE  
 /LOG  
 /TERMINATION\_CHARACTER=^P

---

### Restriction

- The EXTRACT command is used for interactive terminal and DECwindows tests only.
- The /TERMINATION\_CHARACTER qualifier applies to terminal tests only.

---

### Command Parameters

#### *session-file-specification*

Specifies an existing session file. If you enter a file name only, DEC/Test Manager supplies the file type .SESSION. If the session file is in a CMS library, DEC/Test Manager executes a CMS FETCH command for the session file element and deletes the element when done.

#### *input-file-specification*

Specifies the file specification for the input file to be created. If you omit this parameter, DEC/Test Manager uses the session file name and supplies the file type .INP. If the input file is to be placed in a CMS library, DEC/Test

# EXTRACT

Manager executes either a CMS CREATE ELEMENT command or CMS RESERVE and REPLACE commands to place the file in the specified CMS library.

---

## Description

The EXTRACT command extracts an input file from a terminal or DECwindows session file without altering the session file. An input file is a text file you can edit using the text editor of your choice.

See Chapter 8 for information about using the EXTRACT command with terminal session files.

See Chapter 9 for information about using the EXTRACT command with DECwindows session files.

---

## Command Qualifiers

### ***/DECWINDOWS***

Specifies that the session file is a recorded DECwindows session.

### ***/INTERACTIVE (D)***

Specifies that the session file is a recorded terminal session.

### ***/LOG (D)***

### ***/NOLOG***

Controls whether DEC/Test Manager displays informational and success messages on your screen.

### ***/TERMINATION\_CHARACTER=character***

Specifies the character that DEC/Test Manager interprets as the termination character when you extract the input file from the terminal session file. DEC/Test Manager uses this information when translating recording functions in the session file to special strings in the input file.

If you used a termination character other than the default termination character CTRL/P when recording the session file, specify that termination character on the EXTRACT command line.

The termination character can be any single character, such as an asterisk (\*), or a control sequence, such as CTRL/P. To specify a control character, enter a circumflex (^) followed by a letter. For example, to enter the termination character sequence CTRL/D, enter a circumflex followed by a D (/TERMINATION\_CHARACTER=^D). You can also specify a termination character by its ASCII decimal representation. For example, you can use the ASCII number 16 to specify CTRL/P.

---

## Examples

1. DTM> EXTRACT SAMPLE.SESSION SAMPLE.INP  
DTM-S-EXTRACTED, input file SAMPLE.INP created

This example creates the terminal input file SAMPLE.INP. You can edit this file and use it in conjunction with the RECORD/INPUT command.

2. DTM> EXTRACT/DECWINDOWS DECW\_SAMPLE.SESSION DECW\_SAMPLE.INP  
DTM-S-EXTRACTED, input file SAMPLE.INP created

This example creates the DECwindows input file DECW\_SAMPLE.INP. You can edit this file and use it in conjunction with the RECORD/INPUT command.

# FILTER

---

## FILTER

Replaces run-time-dependent information with constants in a specified file. You can use DEC/Test Manager filters on DEC/Test Manager files and other ASCII files.

---

### Format

**FILTER** *file-specification* [/qualifier...]

<b>Command Qualifiers</b>	<b>Default</b>
/ALL	No filters
/DATE	/DATE
/DIRECTORIES	/DIRECTORIES
/FILE_NAMES	/FILE_NAMES
/[NO]LOG	/LOG
/TIME	/TIME
/TRACEBACK	/TRACEBACK
/VERSION	/VERSION

---

### Restrictions

- The **FILTER** command is for use with noninteractive and terminal tests only.

---

### Command Parameter

***file-specification***  
Specifies the file to be filtered.

---

## **Description**

The **FILTER** command enables you to replace run-time-dependent information with constants so that the run-time information does not cause differences when a comparison is performed.

You can use DEC/Test Manager filters on both DEC/Test Manager files and other ASCII files. This command enables you to prepare benchmarks for associations with tests or to see how result files would look if filters were run on them.

You process the file using each of the specified filters. As each specified filter is run on the file, a new version of the file is created; the latest version has all the specified filters run on it. DEC/Test Manager purges all intermediate files.

---

## **Command Qualifiers**

### ***/ALL***

Specifies that all the filters be run on the specified file.

### ***/DATE***

Where the date form is abbreviated, the date filter replaces date stamps by substituting a “d” for each displayed number of the day of the month, an “m” for each displayed letter of the month, and a “y” for each displayed number of the year. Where the date form is spelled out, the month name is replaced by “month”, the numeric day is replaced by “day”, and the year is replaced by “year”.

The following list shows some examples of the date filtering functions; this list is not all inclusive.

17-OCT-1989 with dd-mmm-yyyy  
17 OCT 89 with dd mmm yy  
89.OCT.17 with yy.mmm.dd  
10/17/89 with mm/dd/yy  
1989/10/17 with yyyy/mm/dd  
October 17, 1989 with month day, year  
Oct. 17, 1989 with month day, year  
17.October.1989 with day.month.year

# FILTER

89-October-17 with year-month-day

## ***/DIRECTORIES***

Replaces the directory specification field in the file specification with DISK:[DIRECTORY].

## ***/FILE\_NAMES***

Replaces the file names with FILENAME.EXT.

## ***/LOG (D)***

## ***/NOLOG***

Controls whether DEC/Test Manager displays informational and success messages on your screen.

## ***/TIME***

Replaces time stamps with the following forms:

15:37:53.22 with hh:mm:ss.xxxx

15:37:53 with hh:mm:ss

15:37 with hh:mm

3:37 PM with hh:mm xm

15H37m with hhHmmm

15H37' with hhHmm'

15.37 h with hh.mm h

15 h 37"53 s with hh h mm"ss s

15 h 37 min with hh h mm min

kl 15.37 with kl hh.mm

h 15.37 with h hh.mm

## ***/TRACE\_BACK***

Replaces 8-bit memory addresses with xxxxxxxx.

## ***/VERSION***

Replaces file versions with VERSION.

---

## Example

```
DTM> FILTER/ALL DUA0:[USER01.DTMLIB]FILTER.BMK ""
%DTM-S-FILTERED, expression successfully filtered
```

This example runs the six standard filters on the file FILTER.BMK.

---

## HELP

Displays help text for DEC/Test Manager and Review subsystem commands.

---

### Format

**HELP** *[topic]*

**Command Qualifiers**

None

**Defaults**

None

---

### Command Parameter

*topic*

Specifies a topic about which you want information.

---

### Description

The **HELP** command displays DEC/Test Manager information on your screen. You can access general DEC/Test Manager Help, which provides further information for specific DEC/Test Manager topics, or you can access specific DEC/Test Manager topics directly. If you do not specify a topic parameter, you get a display of available help features and instructions for displaying the text.

---

### Examples

1. DTM> HELP COPY TEST\_DESCRIPTION

This example shows how you can access information about the **COPY TEST\_DESCRIPTION** command from the DEC/Test Manager system.

# HELP

2. DTM\_REVIEW> HELP SHOW/SUMMARY

This example shows how you can access DEC/Test Manager help about the SHOW/SUMMARY command from the DEC/Test Manager Review subsystem.



---

# INSERT GROUP

Places one or more groups in one or more other groups.

---

## Format

**INSERT GROUP** *group-expression1 group-expression2*  
*[/qualifier...] "remark"*

### Command Qualifiers

/[NO]CONFIRM

/[NO]LOG

### Defaults

/NOCONFIRM

/LOG

---

## Restrictions

- DEC/Test Manager does not insert the same group into another group more than once. If the first group is already a subgroup of the second group, DEC/Test Manager informs you that the INSERT operation has already been done.
- DEC/Test Manager does not create recursive group structures. If group B is a subgroup of group A, then group A cannot be a subgroup of group B.

---

## Command Parameters

### *group-expression1*

Specifies the groups to be inserted. These groups become subgroups of the groups in which they are inserted. A group expression can be a group name or a list of group names separated by commas. You can use wildcards.

### *group-expression2*

Specifies the groups into which the subgroups specified in the *group-expression1* parameter are to be inserted. A group expression can be a group name or a list of group names separated by commas. You can use wildcards.

# INSERT GROUP

## *remark*

Specifies a string that contains a comment. You must specify a remark within quotation marks ( " "). The exception to this rule is when you specify a remark string at the remark prompt. If you do not provide a remark string, you are prompted for one. However, a null remark is permitted.

---

## Description

The INSERT GROUP command places one or more groups (subgroups) into one or more other groups. Use this command to create a group hierarchy.

Subsequent changes to tests or subgroups inserted with the INSERT GROUP command are reflected in the group.

See Chapter 6 for more information on groups.

---

## Command Qualifiers

### */CONFIRM*

### */NOCONFIRM (D)*

Controls whether DEC/Test Manager prompts you to confirm each insertion. Valid responses are Yes, No, All, or Quit.

### */LOG (D)*

### */NOLOG*

Controls whether DEC/Test Manager displays informational and success messages on your screen.

---

## Examples

- DTM> INSERT GROUP MARGINS BOUNDARIES  
    Remark: Inserting group MARGINS into group BOUNDARIES  
    %DTM-S-INSERTED, group MARGINS inserted into group BOUNDARIES

This example inserts the MARGINS group into the BOUNDARIES group.

## INSERT GROUP

- DTM> INSERT GROUP RIGHTMARGIN BOUNDARIES,MARGINS  
\_Remark: Inserting RIGHTMARGIN into MARGINS and BOUNDARIES  
%DTM-I-INSERTED, group RIGHTMARGIN inserted into group MARGINS  
%DTM-I-INSERTED, group RIGHTMARGIN inserted into group BOUNDARIES  
%DTM-S-INSERTIONS, 2 insertions completed

This example inserts the RIGHTMARGIN group into both the MARGINS and BOUNDARIES groups.

# INSERT TEST\_DESCRIPTION

---

## INSERT TEST\_DESCRIPTION

Places one or more test descriptions into one or more groups.

---

### Format

```
INSERT TEST_DESCRIPTION test-group-expression  
                        group-expression [/qualifier...]  
                        "remark"
```

#### Command Qualifiers

/[NO]CONFIRM

/[NO]LOG

#### Defaults

/NOCONFIRM

/LOG

#### Parameter Qualifiers

/GROUP

/TEST\_DESCRIPTION

#### Defaults

/TEST\_DESCRIPTION

/TEST\_DESCRIPTION

---

### Command Parameters

#### *test-group-expression*

Specifies the test descriptions or groups of test descriptions that are to be inserted. A test group expression can be a test name, a group name, or a list of these separated by commas. You can use wildcards.

Identify the individual items in the *test-group-expression* as tests or groups using the parameter qualifiers /GROUP and /TEST\_DESCRIPTION.

#### *group-expression*

Specifies the groups into which the test descriptions are to be inserted. A group expression can be a group name or a list of group names separated by commas. You can use wildcards.

# INSERT TEST\_DESCRIPTION

## *remark*

Specifies a string that contains a comment. You must specify a remark within quotation marks ( " "). The exception to this rule is when you specify a remark string at the remark prompt. If you do not provide a remark string, you are prompted for one. However, a null remark is permitted.

---

## Description

The `INSERT TEST_DESCRIPTION` command places one or more test descriptions into one or more groups. DEC/Test Manager will not insert a test description into a group if it is already a member of the group.

You can specify test descriptions in a test group expression, identifying the items in the expression as test names or as group names using the corresponding parameter qualifier. When you specify a test description, it is inserted into the groups specified in the group expression parameter.

When you specify a group name, all the tests that currently belong to that group are inserted into the specified group in their current state. However, the group itself does not become a subgroup of the specified group, and the group name does not become affiliated with the group named in the group expression parameter.

Subsequent changes made to the individual tests or to the contents of the inserted group are not reflected in the group. Use the `INSERT TEST_DESCRIPTION` command with a group name to include the specified subgroup of tests in their current state without having subsequent changes reflected in the final group. See Chapter 6 for more information about using groups.

---

## Command Qualifiers

***/CONFIRM***

***/NOCONFIRM (D)***

Controls whether DEC/Test Manager prompts you to confirm each insertion. Valid responses are Yes, No, All, or Quit.

# INSERT TEST\_DESCRIPTION

***/LOG (D)***

***/NOLOG***

Controls whether DEC/Test Manager displays informational and success messages on your screen.

---

## Parameter Qualifiers

***/GROUP***

Identifies the immediately preceding item in a parameter as a group. If the test group expression is a list, use this qualifier after each item that designates a group. The default is */TEST\_DESCRIPTION*.

***/TEST\_DESCRIPTION***

Identifies the immediately preceding item in a parameter as a test. This is the default.

---

## Example

```
DTM> INSERT TEST_DESCRIPTION MARGINS/GROUP BOUNDARIES
_Remark: Inserting tests in MARGINS into BOUNDARIES
%DTM-I-INSERTED, test description BOTTOMEDGE inserted into group BOUNDARIE
%DTM-I-INSERTED, test description LMARGIN inserted into group BOUNDARIES
%DTM-I-INSERTED, test description RMARGIN inserted into group BOUNDARIES
%DTM-I-INSERTED, test description TOPEdge inserted into group BOUNDARIES
%DTM-S-INSERTIONS, 4 insertions completed
```

This example inserts all the test descriptions in the MARGINS group into the BOUNDARIES group. However, the MARGINS group itself is not inserted.

---

# MODIFY GROUP

Replaces the remark associated with existing groups.

---

## Format

**MODIFY GROUP** *group-expression* [/qualifier...] "remark"

### Command Qualifiers

/[NO]LOG

/[NO]REMARK="string"

### Defaults

/LOG

See text

---

## Command Parameters

### *group-expression*

Specifies the groups to be modified. A group expression can be a group name or a list of group names separated by commas. You can use wildcards.

### *remark*

Specifies a string that contains a comment. You must specify a remark within quotation marks ( " "). The exception to this rule is when you specify a remark string at the remark prompt. If you do not provide a remark string, you are prompted for one. However, a null remark is permitted.

This remark is associated with the MODIFY GROUP command and is logged with it in the history file. It is not the remark that you want to modify for the existing group; you specify that remark with the /REMARK qualifier.

---

## Description

The MODIFY GROUP command replaces the remarks associated with existing groups. You specify the remark you want to modify by using the /REMARK qualifier. You can modify any number of groups with one MODIFY GROUP command.

# MODIFY GROUP

The remark parameter specifies the remark associated with the MODIFY GROUP command and is logged with it in the history file. Because the remark is the only item you can modify for a group, DEC/Test Manager displays an error message if you do not supply a value for the /REMARK qualifier or do not specify the /NOREMARK qualifier which you use to delete any remarks in the specified group.

---

## Command Qualifiers

***/LOG (D)***

***/NOLOG***

Controls whether DEC/Test Manager displays informational and success messages on your screen.

***/REMARK="string"***

***/NOREMARK***

Specifies whether to replace or delete the remark associated with the group. This remark is associated with the group you are modifying; it is not the remark logged with the MODIFY GROUP command.

The /REMARK qualifier adds the specified remark to the group. This string replaces any previous remark associated with the group. The /NOREMARK qualifier deletes the remark from the group.

---

## Example

```
DTM> MODIFY GROUP MARGINS/REMARK="All border tests"  
_Remark: Replacing remark for group MARGINS  
%DTM-S-MODIFIED, group MARGINS modified
```

This example replaces the existing remark associated with the group MARGINS with the remark "All border tests." It also logs the remark "Replacing remark for group MARGINS" with the MODIFY GROUP command in the history file.



---

## MODIFY TEST\_DESCRIPTION

Changes specified field values in existing test descriptions.

---

### Format

**MODIFY TEST\_DESCRIPTION** *test-group-expression*  
*[/qualifier...] "remark"*

#### Command Qualifiers

**/[NO]BENCHMARK=**file-specification

**/[NO]COMMAND=**"DCL-command"

**/[NO]COMPARISON\_TYPE=**keyword

**/[NO]EPILOGUE=**file-specification

**/[NO]FILTER=(**keyword,...)

**/[NO]LOG**

**/[NO]PROLOGUE=**file-specification

**/[NO]REMARK=**"string"

**/[NO]TEMPLATE=**file-specification

**/[NO]VARIABLE=(**variable-name[=variable-value],...)

#### Defaults

Current benchmark

None

None

Current test epilogue

Current filters

/LOG

Current test prologue

Current remark

Current template

Current variables

#### Parameter Qualifiers

**/GROUP**

**/TEST\_DESCRIPTION**

#### Defaults

**/TEST\_DESCRIPTION**

**/TEST\_DESCRIPTION**

---

### Restrictions

- The **/COMMAND** qualifier applies to DECwindows tests only.
- The **/FILTER** qualifier applies to noninteractive and terminal tests only.

# MODIFY TEST\_DESCRIPTION

---

## Command Parameters

### ***test-group-expression***

Specifies the test descriptions to be modified. A test group expression can be a test name, a group name, or a list of these separated by commas. You can use wildcards.

Identify the individual items in the test group expression as tests or groups with the /GROUP and /TEST\_DESCRIPTION parameter qualifiers.

### ***remark***

Specifies a string that contains a comment. You must specify a remark within quotation marks ( " "). The exception to this rule is when you specify a remark string at the remark prompt. If you do not provide a remark string, you are prompted for one. However, a null remark is permitted.

---

## Description

The MODIFY TEST\_DESCRIPTION command changes field values for the test descriptions specified in the test group expression. You can modify all field values except the test name.

Command qualifiers enable you to specify the fields in the test description that are to be changed and also the new values that are to be assigned to the fields. You can modify any number of fields with one MODIFY TEST\_DESCRIPTION command.

The remark parameter is the remark associated with the MODIFY TEST\_DESCRIPTION command and logged with the command in the history file. The /REMARK qualifier is the modified remark to be associated with the modified test description.

---

## Command Qualifiers

### ***/BENCHMARK=file-specification***

### ***/NOBENCHMARK***

Specifies whether the benchmark file should be replaced or disassociated from a test description.

## MODIFY TEST\_DESCRIPTION

The **/BENCHMARK** qualifier adds the specified benchmark file specification to the test description. This benchmark file supersedes any benchmark currently associated with the test description.

The new file specification cannot contain a directory specification that points to the DEC/Test Manager library. However, you can use **DTM\$LIB** without a file name. The new file specification can specify a directory (other than the DEC/Test Manager library) or a CMS library. The new file specification overrides any benchmark directory specified with an earlier **CREATE TEST\_DESCRIPTION** command.

The new file specification need not specify the same type of directory as the file specification you are overriding. For example, if you have been storing your benchmark files in a CMS library, you can specify a directory that is not a CMS library.

If you specify a file name for a benchmark file that does not exist, DEC/Test Manager treats the test like a new test when a collection that contains this test executes. When a benchmark file is created, it will have the file name you specify.

The **/NOBENCHMARK** qualifier removes the benchmark file specification from the test description and replaces it with the default, **test-name.BMK**.

***/COMMAND="DCL-command"***

***/NOCOMMAND***

Specifies a new command to be executed before the recording and execution of a DECwindows test begins. If you specify the **/NOCOMMAND** qualifier, the command is disassociated from the test description. The qualifier applies to DECwindows tests only.

***/COMPARISON\_TYPE=keyword***

Specifies how the result and benchmark files are to be compared. The valid values for keyword are as follows:

<b>Keyword</b>	<b>Meaning</b>
<b>CHARACTERS</b>	Compares files character by character.
<b>RECORDS</b>	Compares files record by record. This is the default for noninteractive terminal tests.

# MODIFY TEST\_DESCRIPTION

<b>Keyword</b>	<b>Meaning</b>
SCREENS	Compares files screen by screen; screens not marked are not compared. This is the default for interactive terminal and DECwindows tests.

DECwindows tests can only use the SCREENS comparison type. The SCREENS comparison type is also the default comparison type for interactive terminal tests. If you specify the /COMPARISON\_TYPE=SCREENS qualifier for a noninteractive test, this value is ignored.

The /NOCOMPARISON\_TYPE qualifier disassociates any comparison type from the test description.

***/EPILOGUE=file-specification***

***/NOEPILOGUE***

Specifies whether the test epilogue file should be replaced or disassociated from a test description.

The /EPILOGUE qualifier adds the specified test epilogue file specification to the test description. The test epilogue file is run whenever the test description is run; it does not affect any collection epilogue file run with the test. The /NOEPILOGUE qualifier removes the current test epilogue file specification from the test description.

***/FILTER=(keyword,...)***

***/NOFILTER=(keyword,...)***

Selects one or more filters to remove run-time variables from the result file produced from the collection run. The following table shows the valid value for keywords:

# MODIFY TEST\_DESCRIPTION

<b>Filter type</b>	<b>Description</b>
<b>ALL</b>	Specifies that all the filters in this table be used
<b>DATE</b>	<p>Where the date form is abbreviated, the date filter replaces date stamps by substituting a "d" for each displayed number of the day of the month, an "m" for each displayed letter of the month, and a "y" for each displayed number of the year. Where the date form is spelled out, the month name is replaced by "month", the numeric day is replaced by "day", and the year is replaced by "year".</p> <p>The following list shows some examples of the date filtering functions; this list is not all inclusive.</p> <p>17-OCT-1989 with dd-mmm-yyyy 17 OCT 89 with dd mmm yy 89.OCT.17 with yy.mmm.dd 10/17/89 with mm/dd/yy 1989/10/17 with yyyy/mm/dd October 17, 1989 with month day, year Oct. 17, 1989 with month day, year 17.October.1989 with day.month.year 89-October-17 with year-month-day</p>
<b>TIME</b>	<p>Replaces time stamps with the following forms:</p> <p>15:37:53.22 with hh:mm:ss.xxxx 15:37:53 with hh:mm:ss 15:37 with hh:mm 3:37 PM with hh:mm xm 15H37m with hhHmmm 15H37' with hhHmm' 15.37 h with hh.mm h 15 h 37"53 s with hh h mm"ss s 15 h 37 min with hh h mm min kl 15.37 with kl hh.mm h 15.37 with h hh.mm</p>
<b>FILE_NAMES</b>	Replaces the file names with FILENAME.EXT
<b>DIRECTORIES</b>	Replaces the directory specification field in the file specification with DISK:[DIRECTORY]
<b>TRACE_BACK</b>	Replaces 8-bit memory addresses with xxxxxxxx
<b>VERSION</b>	Replaces file versions with VERSION

# MODIFY TEST\_DESCRIPTION

If you specify more than one keyword, separate the keywords with commas and enclose the list in parentheses. If you specify only one keyword, omit the parentheses. The **/FILTER** qualifier associates the specified filters with the test description.

The **/NOFILTER** qualifier removes the specified filters from the test description.

DEC/Test Manager does not replace the filters associated with the test description.

***/LOG (D)***

***/NOLOG***

Controls whether DEC/Test Manager displays informational and success messages on your screen.

***/PROLOGUE=file-specification***

***/NOPROLOGUE***

Specifies whether the test prologue file should be replaced or disassociated from the test description.

The **/PROLOGUE** qualifier adds the specified test prologue file specification to the test description. The test prologue is run whenever the test description is run and does not affect any collection prologue run with the test. The **/NOPROLOGUE** qualifier removes the current test prologue file specification from the test description.

***/REMARK="string"***

***/NOREMARK***

Specifies whether to replace or delete the remark associated with the test description.

The **/REMARK** qualifier associates the specified remark with the test description being modified. This string replaces any previous remark. The remark parameter specifies the remark associated with this **MODIFY TEST\_DESCRIPTION** command and is logged with it in the history file.

The **/NOREMARK** qualifier deletes the remark string from the test description.

# MODIFY TEST\_DESCRIPTION

***/TEMPLATE=file-specification***

***/NOTEMPLATE***

Specifies whether to replace or disassociate from the template file. A test description must always have a template file.

***/VARIABLE=(variable-name[=variable-value],...)***

Enables you to associate existing variables with the test description you are modifying. The variables you specify must be defined in the DEC/Test Manager library by using the CREATE VARIABLE command. A variable associated with a test description by this qualifier is local in scope.

The /VARIABLE qualifier also enables you to redefine values for the variables you specify. If you specify an optional value, the variable takes on that value only for this test description; the value of the original variable is unaffected.

If you specify more than one variable name, separate the names with commas and enclose the list in parentheses. If you specify only one variable name, omit the parentheses. You cannot use wildcards.

The /NOVARIABLE qualifier disassociates the specified variables from the test description.

---

## Parameter Qualifiers

***/GROUP***

Identifies the immediately preceding item in the parameter as a group expression. The default is /TEST\_DESCRIPTION. If a test group expression comprises a list, use this qualifier after each item that designates a group.

***/TEST\_DESCRIPTION***

Identifies the immediately preceding item in the parameter as a test expression. This is the default.

# MODIFY TEST\_DESCRIPTION

---

## Examples

1. DTM> MODIFY TEST\_DESCRIPTION RMTEST/BENCHMARK=RMTEST.BMK  
\_Remark: Replacing old benchmark file  
%DTM-S-MODIFIED, test description RMTEST modified.

**This example replaces a benchmark file specification for the test description RMTEST.**

2. DTM> MODIFY TEST\_DESCRIPTION SEND\_MAIL\_TEST -  
\_DTM> /NOBENCHMARK/TEMPLATE=NEWSSENDTEST.COM  
\_Remark: Replacing template file and removing benchmark file  
%DTM-S-MODIFIED, test description SEND\_MAIL\_TEST modified.

**This example replaces the template file specification for the test description SEND\_MAIL\_TEST and removes the existing benchmark file specification.**



---

# MODIFY VARIABLE

Modifies variable definitions in the DEC/Test Manager library.

---

## Format

**MODIFY VARIABLE** *variable-expression* [/qualifier...] "remark"

### Command Qualifiers

/GLOBAL  
 /LOCAL  
 /[NO]LOG  
 /LOGICAL  
 /NUMERIC  
 /[NO]REMARK="string"  
 /STRING  
 /SYMBOL  
 /VALUE=value

### Defaults

Current scope  
 Current scope  
 /LOG  
 Current usage  
 Current type  
 Current remark  
 Current type  
 Current usage  
 Current value

---

## Command Parameters

### *variable-expression*

Specifies the variables to modify. The variable expression can be a variable name or a list of variable names separated by commas. You can use wildcards.

### *remark*

Specifies a string that contains a comment. You must specify a remark within quotation marks (" "). The exception to this rule is when you specify a remark string at the remark prompt. If you do not provide a remark string, you are prompted for one. However, a null remark is permitted.

# MODIFY VARIABLE

---

## Description

The **MODIFY VARIABLE** command changes information about variables in the DEC/Test Manager library. Command qualifiers allow you to change specific information about the variables. You can modify any amount of information with one **MODIFY VARIABLE** command.

Neither numeric symbols nor logical symbols can have a null value. These conditions would generate illegal DCL syntax statements, such as the following:

```
$ variable_name =  
$ define variable_name ""
```

When modifying the type or value of a variable, DEC/Test Manager checks for this illegal condition in the variable fields of all test descriptions.

The remark parameter is the remark associated with the **MODIFY VARIABLE** command and logged with the command in the history file. The **/REMARK** qualifier is the modified remark to be associated with the modified variable.

---

## Command Qualifiers

### ***/GLOBAL***

Makes the variable expression globally accessible. You cannot specify both the **/LOCAL** and **/GLOBAL** qualifiers with the same **MODIFY VARIABLE** command.

### ***/LOCAL***

Makes the variable expression locally accessible. You cannot specify both the **/LOCAL** and **/GLOBAL** qualifiers with the same **MODIFY VARIABLE** command.

### ***/LOG (D)***

### ***/NOLOG***

Controls whether DEC/Test Manager displays informational and success messages on your screen.

# MODIFY VARIABLE

## ***/LOGICAL***

Changes a variable from a VMS symbol to a VMS logical name. You cannot specify both the ***/LOGICAL*** and ***/SYMBOL*** qualifiers with the same **MODIFY VARIABLE** command.

## ***/NUMERIC***

Changes a string variable of a symbol to numeric value. You cannot specify the ***/NUMERIC*** and ***/STRING*** qualifiers with the same **MODIFY VARIABLE** command.

## ***/REMARK="string"***

## ***/NOREMARK***

Determines whether to replace or remove the existing remark.

The ***/REMARK*** qualifier replaces any existing remark with the specified remark. The ***/NOREMARK*** qualifier removes the remark string from the variable, leaving the remark field value null.

## ***/STRING***

Changes a numeric variable of a symbol to a string. You cannot specify the ***/NUMERIC*** and ***/STRING*** qualifiers with the same **MODIFY VARIABLE** command.

## ***/SYMBOL***

Changes a variable from a VMS logical name to a VMS symbol. You must define variables used as symbols as either numeric or string variable types. You cannot specify both the ***/LOGICAL*** and ***/SYMBOL*** qualifiers with the same **MODIFY VARIABLE** command.

## ***/VALUE=value***

Changes the value of the variable. A value specified with the **MODIFY VARIABLE** command replaces any previous value for the variable.

---

## Example

```
DTM> MODIFY VARIABLE INPUT_FILE/VALUE=INPUT.RNO
_Remark: Replacing value of INPUT_FILE with INPUT.RNO
%DTM-S-MODIFIED, variable INPUT_FILE modified.
```

This example assigns a new default value to the variable **INPUT\_FILE**.

# PLAY

---

## PLAY

Executes the specified session file and displays the output on your screen.

---

### Format

**PLAY** *file-specification* [/qualifier...]

#### Command Qualifiers

/COMMAND="DCL-command"

/DECWINDOWS

/[NO]DISPLAY=screen

/INTERACTIVE

/[NO]LOG

/[NO]RESULT\_FILE[=file-specification]

#### Defaults

/INTERACTIVE

See text.

/INTERACTIVE

/LOG

/NORESULT\_FILE

---

### Restriction

- You cannot play a DECwindows session file on a terminal screen.

---

### Command Parameter

#### *file-specification*

Specifies the file containing the interactive terminal or DECwindows session that is to be executed.

---

### Description

The **PLAY** command executes the specified session file and displays the output on your screen. The file is not part of a collection run and the results of the execution (if saved in a file) are not compared. Use qualifiers to select your output file and the screen on which to display the run.

The session file runs as if it were executing on the type of terminal on which it was recorded. If the display terminal has characteristics different from the recording terminal, the playback you see may not appear as you expect it to appear. Even so, the result file is correct regardless of the differences in terminal characteristics.

---

## Command Qualifiers

### ***/COMMAND=DCL-command***

Specifies a command action to be executed before playing a session file.

### ***/DECWINDOWS***

Specifies that the session file being played is a DECwindows session file.

### ***/DISPLAY=screen***

Specifies the display device on which the output is to be displayed. For interactive terminal tests, the default is SYS\$OUTPUT; for DECwindows tests, the default is the DECwindows server, generally indicated by the DECW\$DISPLAY logical name.

### ***/INTERACTIVE (D)***

Specifies that the session file being played is an interactive terminal session file.

### ***/LOG (D)***

### ***/NOLOG***

Controls whether DEC/Test Manager displays informational and success messages on your screen.

### ***/RESULT\_FILE=file-specification***

### ***/NORERESULT\_FILE (D)***

Specifies whether a file is to receive a copy of the output. If you specify */RESULT\_FILE* but do not include a file specification, DEC/Test Manager places the results in a file named file-name.RES. If you specify the */NORERESULT\_FILE* qualifier (the default), the session file is played but the output is not saved.

# PLAY

---

## Example

```
DTM> PLAY MEMO_TEST.SESSION
%DTM-I-BEGIN, your interactive test session is now beginning...
.
.
.
%DTM-S-CONCLUDED, your interactive test session has concluded
DTM>
```

This examples displays a message stating that your interactive terminal session is beginning, executes the specified session file, and displays a message stating that the execution has ended.

---

# RECORD

Records an interactive terminal or DECwindows test.

---

## Format

**RECORD** *test-name* [*/qualifier...*] "*remark*"

Command Qualifiers	Default
/APPEND	None
/[NO]AUTO_COMPARE	See text.
/DISPLAY=screen	See text
/[NO]EPILOGUE	See text
/[NO]FILTERS	See text
/INPUT=file-specification	See text
/KEYSYM=key-symbol	/KEYSYM=Compose/P
/[NO]LOG	/LOG
/[NO]PROLOGUE	See text
/TERMINATION_CHARACTER=character	See text.
/[NO]VARIABLES	See text

---

## Restriction

- The /APPEND qualifier applies to interactive terminal tests only.
- The /FILTER qualifier applies to noninteractive and terminal tests only.

---

## Command Parameters

### *test-name*

Identifies the name of the test to record.

# RECORD

## *remark*

Specifies a string that contains a comment. You must specify a remark within quotation marks (" "). The exception to this rule is when you specify a remark string at the remark prompt. If you do not provide a remark string, you are prompted for one. However, a null remark is permitted.

---

## Description

The RECORD command instructs DEC/Test Manager to record the specified test. If you specify an input file by using the /INPUT qualifier, DEC/Test Manager records from the input file. If you do not use the /INPUT qualifier, DEC/Test Manager records from the terminal or DECwindows workstation.

All activity that occurs in the recording session is stored in the resulting session file.

The RECORD command requires the name of the test to record. You must create this test using the CREATE TEST\_DESCRIPTION command before using the RECORD command.

When you record a DECwindows test, DEC/Test Manager executes a command (if specified) before recording begins.

See Table 3-3 in Chapter 3 for information about the command sequence keys for controlling a recording session.

---

## Command Qualifiers

### ***/APPEND***

Continues recording the specified test after the end of the input file is reached. The /APPEND qualifier applies to interactive terminal tests only.

If you specify an input file using the /INPUT qualifier, DEC/Test Manager records using input from the input file. When the end of the input file is reached, this qualifier enables you to record further; if you do not specify this qualifier, recording is terminated when the end of the input file is reached.



***/AUTO\_COMPARE******/NOAUTO\_COMPARE***

The ***/AUTO\_COMPARE*** and ***/NOAUTO\_COMPARE*** qualifiers determine whether automatic screen comparison is enabled at the start of your interactive terminal recording session; there is no automatic comparisons for DECwindows tests. By default, automatic screen compare is in effect. The ***/NOAUTO\_COMPARE*** qualifier disables automatic screen compare.

***/DISPLAY=screen***

Specifies the display device on which the output is to be displayed. For interactive terminal tests, the default is **SYS\$OUTPUT**; for DECwindows tests, the default is the DECwindows server, generally indicated by the **DECW\$DISPLAY** logical name.

***/EPILOGUE******/NOEPILOGUE (D)***

Specifies whether DEC/Test Manager is to execute the test epilogue file when interactive recording terminates.

***/FILTERS******/NOFILTERS (D)***

Specifies whether DEC/Test Manager is to filter the benchmark file that is produced when interactive recording terminates. When the test is recorded, DEC/Test Manager invokes only the filters specified on the **CREATE TEST\_DESCRIPTION** command.

***/INPUT=file-specification***

Specifies an input file containing a textual representation of an interactive terminal or DECwindows session file. You must create the input file by issuing the **EXTRACT** command on a previously recorded session.

You cannot use wildcards in the file specification.

***/KEYSYM***

Specifies the key symbol that is associated with a key for use in controlling DECwindows test recording sessions. The default key symbol is **0xFFE9**, which is the Compose Character key.

# RECORD

To respecify the keySYM key, enter its decimal or hexadecimal encoding. For example, to use CTRL as the command key symbol, enter the following command:

```
DTM> RECORD test-name/KEYSYM=0xFFE3
```

The command keySYM key must be in the DECwindows Latin-1 keySYM encodings. Display the file DECW\$INCLUDE:KEYSYMDEF.H with the DCL TYPE command to view a listing of the Latin-1 keySYM keys.

## ***/LOG (D)***

### ***/NOLOG***

Controls whether DEC/Test Manager displays informational and success messages on your screen.

## ***/PROLOGUE***

### ***/NOPROLOGUE (D)***

Specifies whether DEC/Test Manager is to execute the test prologue file before interactive recording begins.

## ***/TERMINATION\_CHARACTER=character***

Specifies a character sequence for interactive terminal sessions that, when pressed twice, terminates the recording of an interactive terminal or DECwindows session. The default termination character is the sequence CTRL/P. When pressed once, CTRL/P temporarily suspends the recording session to introduce a recording function. For example, CTRL/P E (end automatic screen comparison) is the recording function instructing DEC/Test Manager to terminate automatic screen comparison and to begin manual screen comparison.

The termination character can be any single character, such as an asterisk (\*), or a control sequence, such as CTRL/P. To specify a control character, enter a circumflex (^) followed by a letter. For example, to enter the termination character sequence CTRL/D, enter a circumflex followed by a D (/TERMINATION\_CHARACTER=^D). You can also specify a termination character by its ASCII decimal representation. For example, you can use ASCII number 16 to specify CTRL/P.

On interactive terminal tests, you can also terminate the recording session by entering the DCL LOGOUT command. If you do not want an accounting summary, enter the DCL STOP/IDENTIFICATION=0 command.

***/VARIABLES******/NOVARIABLES (D)***

Specifies whether DEC/Test Manager is to associate local variables with the test before interactive testing begins.

---

**Example**

```
DTM> RECORD MAIL_TEST
_Remark: Going to Record the MAIL facility
%DTM-I-BEGIN, your interactive test session is now beginning...
Type CTRL/P twice to terminate the session.

$
.
.
.
^P

%DTM-I-BMK_SAVED, benchmark has been saved in file DUA1:[USER01.DTMLIB]MAIL_TEST.BMK;1
%DTM-S-RECORDED, test MAIL_TEST has been successfully recorded in file
DUA1:[USER01.DTMLIB]MAIL_TEST.SESSION
DTM>
```

**This example shows the recording of interactive terminal test MAIL\_TEST.**

# RECREATE

---

## RECREATE

Re-creates a collection, providing the same attributes that were part of the original collection.

---

### Format

**RECREATE** *collection-name* [/qualifier...] "remark"

Command Qualifiers	Defaults
/[NO]CONFIRM	/CONFIRM
/[NO]LOG	/LOG

---

### Command Parameters

***collection-name***

Specifies the test collection that is to be re-created.

***remark***

Specifies a string that contains a comment. You must specify a remark within quotation marks (" "). The exception to this rule is when you specify a remark string at the remark prompt. If you do not provide a remark string, you are prompted for one. However, a null remark is permitted.

This remark is logged with the RECREATE command in the history file. It is not associated with the re-created collection.

---

### Description

The RECREATE command re-creates a collection. When you create a collection, DEC/Test Manager stores the command in the history. When you re-create a collection using the RECREATE command, DEC/Test Manager uses the command stored in the history so that none of the file specifications or qualifiers are changed.

# RECREATE

DEC/Test Manager deletes the old collection and all files related to it (except the benchmark file) from the DEC/Test Manager library and then creates a new collection with the latest available versions of each required file.

You issue the RECREATE command to have your collection reflect changes made to a test, variable definitions, or something else in the library since you originally created the collection. You might also use this command if you subsequently created more tests to be included in the collection.

The RECREATE command cannot re-create a collection that you have deleted.

## CAUTION

Ensure that the prologue and epilogue files associated with the collection and template, prologue, and epilogue files associated with the tests in the collection exist in their proper directories. If DEC/Test Manager cannot find the template, prologue, or epilogue files specified on the original CREATE TEST\_DESCRIPTION command, the original collection is deleted but a new collection is not created.

---

## Command Qualifiers

***/CONFIRM (D)***

***/NOCONFIRM***

Controls whether DEC/Test Manager prompts you to confirm that you want to re-create a collection. Valid responses are Yes, No, All, or Quit.

***/LOG (D)***

***/NOLOG***

Controls whether DEC/Test Manager displays informational and success messages on your screen.

# RECREATE

---

## Example

```
DTM> RECREATE MAIL_COLL
_Remark: Re-creating Collection MAIL_COLL
Collection MAIL_COLL has not been reviewed, confirm recreation:[Y/N] (N) Y

%DTM-S-DELETED, collection MAIL_COLL deleted
%DTM-S-CREATED, collection MAIL_COLL created
%DTM-S-RECREATED, collection MAIL_COLL has been re-created
```

**This example re-creates collection MAIL\_COLL. In this example, DEC/Test Manager prompts for confirmation to re-create the collection without first reviewing it.**

---

## REMARK

Places a remark in the DEC/Test Manager history file.

---

### Format

**REMARK** "*remark*" [/*qualifier*]

Command Qualifier	Default
/[NO]LOG	/LOG

---

### Command Parameter

***remark***

Specifies a string that contains a comment. A remark must be specified within quotation marks (" "). If you do not provide a remark string, you are prompted for one, but a null remark is permitted.

---

### Description

The **REMARK** command adds a remark to the DEC/Test Manager history file. If you allow DEC/Test Manager to prompt you for a remark, the remark can be up to 254 characters. Otherwise, the total command line length must follow VMS conventions. The remark is recorded in the history file in the following format:

```
date time user-name REMARK "remark"
```

Use this command to enter a comment about an unusual occurrence in the DEC/Test Manager library.

# REMARK

---

## Command Qualifier

***/LOG (D)***

***/NOLOG***

Controls whether DEC/Test Manager displays informational and success messages on your screen.

---

## Example

```
DTM> REMARK "Correcting problems created by -  
_DTM> system crash while collection was running"  
%DTM-S-REMARK, remark added to history file
```

This example adds the remark "Correcting problems created by system crash while collection was running." You might enter such a remark into your history file to clarify your reason for entering a **VERIFY/RECOVER** command to correct problems with the library.



---

# REMOVE GROUP

Removes one or more groups from one or more other groups.

---

## Format

**REMOVE GROUP** *group-expression1 group-expression2*  
[/*qualifier...*] "*remark*"

### Command Qualifiers

/[NO]CONFIRM

/[NO]LOG

### Defaults

/CONFIRM

/LOG

---

## Command Parameters

### *group-expression1*

Specifies the groups to be removed. A group expression can be a group name or a list of group names separated by commas. You can use wildcards.

### *group-expression2*

Specifies the groups from which the specified groups are to be removed. A group expression can be a group name or a list of group names separated by commas. You can use wildcards.

### *remark*

Specifies a string that contains a comment. You must specify a remark within quotation marks (" "). The exception to this rule is when you specify a remark string at the remark prompt. If you do not provide a remark string, you are prompted for one. However, a null remark is permitted.

# REMOVE GROUP

---

## Description

The REMOVE GROUP command removes one or more groups from one or more other groups, thereby reversing the action of the INSERT GROUP command. Removing a group does not delete the group from the DEC/Test Manager library; it changes the group hierarchy by canceling a group's connection as a subgroup of another group.

---

## Command Qualifiers

***/CONFIRM (D)***

***/NOCONFIRM***

Controls whether DEC/Test Manager prompts you to confirm each removal. Valid responses are Yes, No, All, or Quit.

***/LOG (D)***

***/NOLOG***

Controls whether DEC/Test Manager displays informational and success messages on your screen.

---

## Examples

1. DTM> REMOVE GROUP LEFTMARGIN MARGINS /NOCONFIRM

    Remark: Removing LEFTMARGIN from MARGIN

    %DTM-S-REMOVED, group LEFTMARGIN removed from group MARGINS

**This example removes the LEFTMARGIN group from the MARGINS group.**

2. DTM> REMOVE GROUP RIGHTMARGIN MARGINS,BOUNDARIES /CONFIRM

    Remark: Removing RIGHTMARGIN from MARGINS and BOUNDARIES

    Confirm removal of group RIGHTMARGIN from group BOUNDARIES [Y/N] (N) Y

    %DTM-I-REMOVED, group RIGHTMARGIN removed from group BOUNDARIES

    Confirm removal of group RIGHTMARGIN from group MARGINS [Y/N] (N) Y

    %DTM-I-REMOVED, group RIGHTMARGIN removed from group MARGINS

    %DTM-S-REMOVALS, 2 removals completed

**This example removes the RIGHTMARGIN group from the BOUNDARIES group and the MARGINS group and prompts you to confirm each transaction.**

---

## REMOVE TEST\_DESCRIPTION

Removes one or more test descriptions from one or more groups.

---

### Format

```
REMOVE TEST_DESCRIPTION test-group-expression  
group-expression [/qualifier...]  
"remark"
```

#### Command Qualifiers

/[NO]CONFIRM

/[NO]LOG

#### Defaults

/CONFIRM

/LOG

#### Parameter Qualifiers

/GROUP

/TEST\_DESCRIPTION

#### Defaults

/TEST\_DESCRIPTION

/TEST\_DESCRIPTION

---

### Command Parameters

#### *test-group-expression*

Specifies the test descriptions to be removed. A test group expression can be a test name, a group name, or a list of these separated by commas. You can use wildcards.

Identify the individual items in the test group expression as tests or groups with the /GROUP and /TEST\_DESCRIPTION parameter qualifiers.

#### *group-expression*

Specifies the group from which the test descriptions are to be removed. A group expression can be a group name or a list of group names separated by commas. You can use wildcards.

# REMOVE TEST\_DESCRIPTION

## *remark*

Specifies a string that contains a comment. You must specify a remark within quotation marks (" "). The exception to this rule is when you specify a remark string at the remark prompt. If you do not provide a remark string, you are prompted for one. However, a null remark is permitted.

---

## Description

The REMOVE TEST\_DESCRIPTION command removes one or more test descriptions from one or more groups, thereby reversing the action of the INSERT TEST\_DESCRIPTION command. It does not delete the test description from the DEC/Test Manager library.

---

## Command Qualifiers

### */CONFIRM (D)*

### */NOCONFIRM*

Controls whether DEC/Test Manager displays each test description name for you to confirm removal of the test description. Valid responses are Yes, No, All, or Quit.

### */LOG (D)*

### */NOLOG*

Controls whether DEC/Test Manager displays informational and success messages on your screen.

---

## Parameter Qualifiers

### */GROUP*

Identifies the immediately preceding item in the parameter as a group expression. If a test group expression is a list, use this qualifier after each item that designates a group. The default is /TEST\_DESCRIPTION.

### */TEST\_DESCRIPTION*

Identifies the immediately preceding item in the parameter as a test expression. This is the default.

# REMOVE TEST\_DESCRIPTION

---

## Example

```
DTM> REMOVE TEST_DESCRIPTION MARGINS/GROUP BOUNDARIES/NOCONFIRM
_Remark: Removing MARGINS tests from BOUNDARIES
%DTM-I-REMOVED, test_description BOTTOMEDGE removed from group BOUNDARIES
%DTM-I-REMOVED, test_description LEFTMARGIN removed from group BOUNDARIES
%DTM-I-REMOVED, test_description RIGHTMARGIN removed from group BOUNDARIES
%DTM-I-REMOVED, test_description TOPEdge removed from group BOUNDARIES
%DTM-S-REMOVALS, 4 removals completed
```

**This example removes all the tests in the MARGINS group from the BOUNDARIES group without prompting for confirmation.**

# RESTORE

---

## RESTORE

Converts a DECwindows input file to a DECwindows session file.

---

### Format

**RESTORE** *input-file-specification* [*session-file-specification*]  
[*/qualifier...*]

**Command Qualifiers**

/DECWINDOWS

**Defaults**

/DECWINDOWS

---

### Restriction

- The RESTORE command is used for DECwindows test files only.

---

### Command Parameters

***input-file-specification***

Specifies an existing input file. If you enter a file name only, DEC/Test Manager supplies the file type .INP. If the input file is in a CMS library, DEC/Test Manager executes a CMS FETCH command for the session file element and deletes the element when done.

***session-file-specification***

Specifies a session file to create. If you omit this parameter, DEC/Test Manager uses the session file name and supplies the file type .SESSION. If the session file is to be placed in a CMS library, DEC/Test Manager executes either a CMS CREATE ELEMENT command or CMS RESERVE and REPLACE commands to place the file in the specified CMS library.

---

## Description

The **RESTORE** command converts a DECwindows input file into binary session format. DEC/Test Manager uses the binary file to play back the session.

If you have made an error in editing the input file, DEC/Test Manager issues an error message with the line number of the error; a session file is created but it will be incomplete or with errors.

See Chapter 9 for information about using the **RESTORE** command with DECwindows session files.

---

## Command Qualifiers

### ***/DECWINDOWS***

Specifies that the file to be translated is a DECwindows input file.

---

## Example

```
DTM> RESTORE/DECWINDOWS SAMPLE.INP SAMPLE.SESSION  
DTM-S-RESTORED, input file SAMPLE.SESSION created
```

This example translates the input file **SAMPLE.INP** to the session file **SAMPLE.SESSION**.

# REVIEW

---

## REVIEW

Invokes the Review subsystem to examine and manipulate test results for a collection of tests.

---

### Format

**REVIEW** *collection-name* [/qualifier] ["remark"]

**Command Qualifier**

/[NO]READ\_ONLY

**Default**

/NOREAD\_ONLY

---

### Restriction

- You can only review noninteractive and interactive terminal tests in the Review subsystem. DECwindows tests must be reviewed through the DEC/Test Manager DECwindows user interface (see Chapter 5).

---

### Command Parameters

***collection-name***

Identifies a collection created with the CREATE COLLECTION command. You cannot use wildcards when specifying the collection name parameter.

***remark***

Specifies a string that contains a comment. This remark is optional.

---

### Description

When you specify the REVIEW command at the DEC/Test Manager prompt (or DTM REVIEW at the DCL prompt), you invoke the Review subsystem, signified by the DTM\_REVIEW> prompt. The Review subsystem enables you to examine and manipulate test results for the specified collection; DEC/Test Manager displays the Collection Summary Information.



# REVIEW

You must run and compare a collection before you can review it. A summary of the results for each test description is stored in the result description for each test. A result description name for a test is the same as its test name.

If you enter the **REVIEW** command with the **/NOREAD\_ONLY** qualifier, DEC/Test Manager identifies you as the primary reviewer of the collection. Only one person at a time can be the primary reviewer. This reviewer can use all Review subsystem commands.

If you specify the **/READ\_ONLY** qualifier, you are designated as a read-only reviewer of the collection. DEC/Test Manager permits multiple read-only reviewers for a collection. You can browse through the result descriptions and print files, but you cannot make any changes to the result descriptions. Read-only reviewers cannot enter the **UPDATE** or **INSERT** commands.

The primary reviewer exercises control over the collection being reviewed. When a read-only reviewer reviews a collection, the collection summary information that is being reviewed (using the **SHOW** command) can be inaccurate if the primary reviewer is currently reviewing and manipulating the collection data. As a consequence, print jobs may seem to disappear if the primary reviewer deletes the file marked for printing before you exit from the Review subsystem (the print command executes when you leave the Review subsystem).

You can reduce the probability that a file marked for printing will disappear by using the **PRINT** command with the **/NOW** or **/SELECTED** qualifier. The **/NOW** qualifier causes DEC/Test Manager to queue for printing only the files specified on the current command line. The **/SELECTED** qualifier causes DEC/Test Manager to queue for printing all files specified during the current review session, including those specified on the current command line.

See Chapter 5 for more information about the Review subsystem. See Command Dictionary for more information about the Review subsystem commands.

# REVIEW

---

## Command Qualifiers

***/READ\_ONLY***  
***/NOREAD\_ONLY (D)***

Determines whether you are a primary or read-only user of the Review subsystem. There can be only one primary reviewer at a time. This reviewer can enter all Review subsystem commands. There can be multiple read-only reviewers who cannot make changes to the result descriptions or to the files they describe. Read-only reviewers cannot enter the INSERT or UPDATE commands.

---

## Example

```
DTM> REVIEW MAIL_COLL
Collection MAIL_COLL with 7 tests was created on 7-SEP-1989 09:30:28 by th
command:
      CREATE COLLECTION MAIL_COLL MAIL* "Creating the MAIL tests collect
      Last Review Status = not previously reviewed
      Success count = 4
      Unsuccessful count = 1
      New test count = 2
      Updated test count = 0
      Comparisons aborted = 0
      Test not run count = 0

Result Description MAIL_TEST      Comparison Status : Unsuccessful

DTM_REVIEW>
```

This example invokes the Review subsystem for the collection MAIL\_COLL.

---

# RUN

Executes a collection interactively producing result files that are subsequently compared with benchmark files.

---

## Format

**RUN** *collection-name* [/qualifier...] "*remark*"

### Command Qualifiers

/[NO]CONFIRM

/[NO]LOG

/[NO]LOG\_FILE=file-specification

/[NO]OUTPUT=file-specification

### Defaults

/CONFIRM

/LOG

LOG\_FILE=SYS\$OUTPUT

/OUTPUT=SYS\$OUTPUT

---

## Command Parameters

### *collection-name*

Identifies a collection created with the CREATE COLLECTION command.

### *remark*

Specifies a string that contains a comment. You must specify a remark within quotation marks (" "). The exception to this rule is when you specify a remark string at the remark prompt. If you do not provide a remark string, you are prompted for one. However, a null remark is permitted.

---

## Description

The RUN command executes a collection of tests interactively by processing test template files or previously recorded session files. This produces a result file for each test in the collection; these results are subsequently compared to their respective benchmark files.

# RUN

When you issue the RUN command, your screen displays the output of the noninteractive and interactive terminal test templates as they are put into the result file; you are also shown any informational messages that are generated. You can use the /OUTPUT qualifier to redirect this output to a file, or you can suppress it with the /NOOUTPUT qualifier.

Regardless of the qualifier you use, the RUN command occupies your terminal or workstation while the collection executes. Therefore, when you do not need to see the output from your test, execute collections in batch using the DEC/Test Manager SUBMIT command instead of the RUN command. (See the SUBMIT command for more information on executing your collection in batch.)

## NOTE

If, instead of using the SUBMIT command, you submit a command procedure that includes the RUN command, you must specify /NOLOG\_FILE or /LOG\_FILE=file-specification, where the file specification specifies something other than SYS\$OUTPUT. If you specify SYS\$OUTPUT with the /LOG\_FILE qualifier, the RUN command attempts to write to your batch job's log file. The VMS operating system does not allow this file to be shared, and your RUN command will fail.

If you specify a test name rather than a collection name as the parameter to the RUN command, DEC/Test Manager prompts you to confirm that you want to create a collection containing the test. If you type YES, DEC/Test Manager creates a collection with the same collection name as the specified test name—if a collection by that name does not already exist—and executes this collection.

Press CTRL/C to abort a RUN command. Pressing CTRL/C allows DEC/Test Manager to clean up and restore the library to a consistent state. Do not press CTRL/Y to abort a collection run.

The DEC/Test Manager RUN command is different from the DCL RUN command; you cannot use the DCL RUN command to execute a collection of tests.

---

## Command Qualifiers

***/CONFIRM (D)***

***/NOCONFIRM***

Controls whether DEC/Test Manager prompts for confirmation when you rerun a collection. Valid responses are Yes, No, All, or Quit.

***/LOG (D)***

***/NOLOG***

Controls whether DEC/Test Manager displays informational and success messages on your screen.

***/LOG\_FILE=file-specification***

***/NOLOG\_FILE***

Specifies whether a log file for the collection is to be created and where the contents of the log file is to be displayed.

This file is similar in content to the log file created when you issue the DEC/Test Manager SUBMIT command to run a collection of tests in batch mode. The log file contains output the test run generates other than the output from the test itself, such as output the prologue and epilogue files generate.

The */LOG\_FILE* qualifier creates a log file in the specified file. The output is directed to SYS\$OUTPUT. The */NOLOG\_FILE* qualifier suppresses the creation of a log file.

***/OUTPUT=file-specification***

***/NOOUTPUT***

Specifies where a copy of the output of the RUN command (the test result files) is displayed or written.

The */OUTPUT* qualifier directs the output to SYS\$OUTPUT or to the specified file or device. The */NOOUTPUT* qualifier suppresses output.

# RUN

---

## Example

```
DTM> RUN SEND_MAIL_TEST "simple test of SEND_MAIL_TEST command"
Starting SEND_MAIL_TEST test run...

%DTM-I-BEGIN, your interactive test session is now beginning...
.
.
%DTM-S-CONCLUDED, your interactive test session has concluded

Performing post-run cleanup with comparison...

DTM-I-SUCCEEDED, the comparison for test SEND_MAIL_TEST succeeded
DTM-S-COMPARED, collection SEND_MAIL_TEST compared
DTM>
```

This example executes the collection `SEND_MAIL_TEST` interactively.

---

## SET BENCHMARK\_DIRECTORY

Specifies the default directory that DEC/Test Manager searches for benchmark files.

---

### Format

**SET BENCHMARK\_DIRECTORY** *[directory-specification]*  
*[/qualifier] "remark"*

**SET NOBENCHMARK\_DIRECTORY** *["remark"]*

Command Qualifier	Default
/[NO]LOG	/LOG

---

### Command Parameters

#### *directory-specification*

Specifies the directory that DEC/Test Manager searches for benchmark files. Unless instructed otherwise, DEC/Test Manager places updated benchmark files in this directory. This directory may be a CMS library.

#### *remark*

Specifies a string that contains a comment. You must specify a remark within quotation marks (" "). The exception to this rule is when you specify a remark string at the remark prompt. If you do not provide a remark string, you are prompted for one. However, a null remark is permitted.

---

### Description

The SET BENCHMARK\_DIRECTORY command instructs DEC/Test Manager to establish the specified directory as the default directory for benchmark files for the current DEC/Test Manager library. DEC/Test Manager searches this directory for benchmark files and places new and updated benchmark files in this directory.

## SET BENCHMARK\_DIRECTORY

When you create a collection with the `CREATE COLLECTION` command and the `/BENCHMARK_DIRECTORY` qualifier and specify a default benchmark directory for the collection, DEC/Test Manager references that directory for benchmark files associated with tests in that collection. If you associate a directory specification with the benchmark file for a specific test, DEC/Test Manager references that directory rather than either the default benchmark directory for the library or the collection benchmark directory when executing that test.

If you store your benchmark files in a CMS library, you can use classes to specify which version of the benchmark file you want DEC/Test Manager to access. This allows you to select from several versions of a single benchmark file, which is useful if you are maintaining several base levels of a program.

The `SET NOBENCHMARK_DIRECTORY` command clears any library default benchmark directory established with the `SET BENCHMARK_DIRECTORY` command.

---

### Command Qualifier

*/LOG (D)*

*/NOLOG*

Controls whether DEC/Test Manager displays informational and success messages on your screen.

---

### Example

```
DTM> SET BENCHMARK_DIRECTORY DUA0:[USER01.BMK]
_Remark: New default benchmark directory
%DTM-S-NEWDEF, DUA0:[USER01.BMK] is the new default collection
benchmark directory
DTM>
```

This example establishes a new default benchmark directory for the library.



---

## SET EPILOGUE

Establishes the specified file as the default collection epilogue file for all subsequently created collections.

---

### Format

**SET EPILOGUE** *file-specification* [/qualifier] "remark"

**SET NOEPILOGUE** "remark"

Command Qualifier	Default
/[NO]LOG	/LOG

---

### Command Parameters

***file-specification***

Specifies the file as the default collection epilogue file.

***remark***

Specifies a string that contains a comment. You must specify a remark within quotation marks ( " "). The exception to this rule is when you specify a remark string at the remark prompt. If you do not provide a remark string, you are prompted for one. However, a null remark is permitted.

---

### Description

The **SET EPILOGUE** command establishes the default collection epilogue file for subsequently created collections. DEC/Test Manager verifies the existence of the collection epilogue when you create a collection. This file may be in a CMS library.

You can override the default when you create a collection by assigning another file with the **CREATE COLLECTION/EPILOGUE** command.

The **SET NOEPILOGUE** command cancels a previously established default collection epilogue file.

# SET EPILOGUE

---

## Command Qualifier

*/LOG (D)*

*/NOLOG*

Controls whether DEC/Test Manager displays informational and success messages on your screen.

---

## Examples

1. DTM> SET EPILOGUE DUA0:[USER01.DTMEPI]EPILOGUE.COM ""  
%DTM-S-NEWDEF, DUA0:[USER01.DTMEPI]EPILOGUE.COM is the new default  
collection epilogue

This example specifies a file as the default collection epilogue file for subsequently created collections.

2. DTM> SET NOEPILOGUE "canceling epilogue"  
DTM-S-DEFCANCEL, default epilogue canceled

This example cancels a default collection epilogue file.

---

# SET LIBRARY

Selects an existing DEC/Test Manager library.

---

## Format

**SET LIBRARY** *directory-specification* [/qualifier...]

Command Qualifiers	Defaults
/[NO]LOG	/LOG
/[NO]VERIFY	/VERIFY

---

## Command Parameter

***directory-specification***

Specifies a directory that has been defined as a DEC/Test Manager library with the CREATE LIBRARY command.

---

## Description

The SET LIBRARY command selects an existing DEC/Test Manager library for use with DEC/Test Manager commands. This library is your current library and remains your current library until you select another library with the CREATE LIBRARY or SET LIBRARY command, or until you log out.

DEC/Test Manager issues an error message if you use the SET LIBRARY command with a directory specification that is not a DEC/Test Manager library.

# SET LIBRARY

---

## Command Qualifiers

***/LOG (D)***

***/NOLOG***

Controls whether DEC/Test Manager displays informational and success messages on your screen.

***/VERIFY (D)***

***/NOVERIFY***

Specifies whether DEC/Test Manager is to verify that the specified directory is a valid DEC/Test Manager library.

The */VERIFY* qualifier causes DEC/Test Manager to verify that the specified directory is a valid DEC/Test Manager library before selecting it as the current DEC/Test Manager library. If the specified directory is not a valid library, DEC/Test Manager issues an error message.

The */NOVERIFY* qualifier causes DEC/Test Manager to select the library without verifying that it is a valid DEC/Test Manager library. You must specify a valid DEC/Test Manager library.

---

## Example

```
DTM> SET LIBRARY [.DTMLIB]
%DTM-S-LIBIS, DTM library is DUA0:[USER01.DTMLIB]
```

This example selects a DEC/Test Manager library.

---

## SET PROLOGUE

Establishes the specified file as the default collection prologue file for all subsequently created collections.

---

### Format

**SET PROLOGUE** *file-specification* [/qualifier] "remark"

**SET NOPROLOGUE** "remark"

Command Qualifier	Default
/[NO]LOG	/LOG

---

### Command Parameters

#### *file-specification*

Specifies the file to be the default collection prologue file.

#### *remark*

Specifies a string that contains a comment. You must specify a remark within quotation marks (" "). The exception to this rule is when you specify a remark string at the remark prompt. If you do not provide a remark string, you are prompted for one. However, a null remark is permitted.

---

### Description

The SET PROLOGUE command establishes the specified file as the default prologue file for subsequently created collections. DEC/Test Manager verifies the existence of the collection prologue when you create a collection. This file may be in a CMS library.

You can override the default when you create a collection by assigning another file with the CREATE COLLECTION/PROLOGUE command.

The SET NOPROLOGUE command cancels a previously established default collection prologue file.

# SET PROLOGUE

---

## Command Qualifier

***/LOG (D)***

***/NOLOG***

Controls whether DEC/Test Manager displays informational and success messages on your screen.

---

## Examples

1. DTM> SET PROLOGUE DUA0:[USER01.DTMPRO]PROLOGUE.COM ""  
%DTM-S-NEWDEF, DUA0:[USER01.DTMPRO]prologue.com is the new default  
collection prologue.

**This example specifies a default collection prologue file.**

2. DTM> SET NOPROLOGUE "canceling prologue"  
DTM-S-DEFCANCEL, Default prologue canceled

**This example cancels a default collection prologue file.**

---

## SET TEMPLATE\_DIRECTORY

Establishes the default directory that DEC/Test Manager searches for test template files.

---

### Format

**SET TEMPLATE\_DIRECTORY** *directory-specification* [/qualifier]  
"remark"

**SET NOTEMPLATE\_DIRECTORY** "remark"

Command Qualifier	Default
[/NO]LOG	/LOG

---

### Command Parameters

***directory-specification***

Specifies the directory that DEC/Test Manager searches for template files. Unless otherwise instructed, DEC/Test Manager uses this directory. This directory may be a CMS library.

***remark***

Specifies a string that contains a comment. You must specify a remark within quotation marks (" "). The exception to this rule is when you specify a remark string at the remark prompt. If you do not provide a remark string, you are prompted for one. However, a null remark is permitted.

---

### Description

The SET TEMPLATE\_DIRECTORY command instructs DEC/Test Manager to establish the specified directory as the default directory for template files in the current DEC/Test Manager library.

# SET TEMPLATE\_DIRECTORY

When you create a collection using the `/TEMPLATE_DIRECTORY` qualifier, DEC/Test Manager references that directory for templates associated with tests in that collection. If you include a directory specification in the template file specification for a test in the collection, DEC/Test Manager references that directory rather than either the default template directory or the collection template directory when executing that test.

If the directory you specify is a CMS library, DEC/Test Manager issues the appropriate CMS commands to remove, replace, or insert files in the library. These CMS commands are logged in the DEC/Test Manager history file.

If you store your template files in a CMS library, you can use classes to specify which version of the template file you want DEC/Test Manager to access. This allows you to select from several versions of a single template file, which is useful if you are maintaining several base levels of a program.

The `SET NOTEMPLATE_DIRECTORY` command clears any library default template directory established with the `SET TEMPLATE_DIRECTORY` command.

---

## Command Qualifier

***/LOG (D)***

***/NOLOG***

Controls whether DEC/Test Manager displays informational and success messages on your screen.

---

## Example

```
DTM> SET TEMPLATE_DIRECTORY DUA0:[USER01.TEMPLATE]
_Remark: New default template directory
%DTM-S-NEWDEF, DUA0:[USER01.TEMPLATE] is the new default collection
template directory
DTM>
```

This example establishes a new default template directory for the library.



---

## SHOW ALL

Displays a summary describing the current library.

---

### Format

**SHOW ALL** *[/qualifier]*

**Command Qualifier**

*/OUTPUT[=file-specification]*

**Default**

*/OUTPUT=SYS\$OUTPUT*

---

### Description

The SHOW ALL command displays the current directory specifications for the directories or CMS libraries containing your epilogue, prologue, benchmark and template files, and the number of collections, test descriptions, groups, and variables in the library.

---

### Command Qualifier

***/OUTPUT[=file-specification]***

Sends the SHOW ALL information to the specified file. If you specify the file name but omit the file type, the file type defaults to .LIS. The output is sent to SYS\$OUTPUT.

# SHOW ALL

---

## Example

```
DTM> SHOW ALL
```

```
Description of DEC/Test Manager Library DBA1$:[project.dsrlib]
```

```
Default template directory: DUA0:[USER01.TEMP] "Default template library"  
Default benchmark directory: DUA0:[USER01.BMK] "Default benchmark library"  
Default collection prologue: DUA0:[USER01.PROEPILIB]PROLOGUE_1.COM  
Default collection epilogue: DUA0:[USER01.PROEPILIB]EPILOGUE_1.COM  
Number of collections:      25  
Number of test descriptions: 48  
Number of groups:          8  
Number of variables:       7  
DTM>
```

**This example describes the current DEC/Test Manager library.**

# SHOW BENCHMARK\_DIRECTORY

---

## SHOW BENCHMARK\_DIRECTORY

Displays the directory specification for the current default benchmark directory.

---

### Format

**SHOW BENCHMARK\_DIRECTORY** *[/qualifier]*

**Command Qualifier**

/OUTPUT[=file-specification]

**Default**

/OUTPUT=SYS\$OUTPUT

---

### Description

The SHOW BENCHMARK\_DIRECTORY command displays the directory specification for the default directory for benchmark files. This directory can be a CMS library.

---

### Command Qualifier

*/OUTPUT[=file-specification]*

Sends the requested information to a specified file. If you specify the file name but omit the file type, the file type defaults to .LIS. The output is sent to SYS\$OUTPUT.

---

### Example

```
DTM> SHOW BENCHMARK_DIRECTORY
      DUA0:[USER01.BMK]                "Default benchmark library"
DTM>
```

This command displays the directory specification for the current default benchmark directory.

# SHOW COLLECTION

---

## SHOW COLLECTION

Displays information about collections in the current DEC/Test Manager library.

---

### Format

**SHOW COLLECTION** *collection-expression* [/qualifier...]

Command Qualifiers	Defaults
/BENCHMARK_DIRECTORY	None
/BRIEF	/INTERMEDIATE
/CLASS=(keyword=class-name,...)	None
/FULL	/INTERMEDIATE
/INTERMEDIATE	/INTERMEDIATE
/OUTPUT[=file-specification]	/OUTPUT=SYS\$OUTPUT
/TEMPLATE_DIRECTORY	None

---

### Command Parameter

***collection-expression***

Specifies the collections about which information is to be displayed. A collection expression can be a collection name or a list of collection names separated by commas. You can use wildcards.

---

### Description

The SHOW COLLECTION command displays information about collections in the current DEC/Test Manager library. These collections include currently running collections, collections that have been created but not run, collections that have been run but not deleted, and collections that have been run and stopped (partially run collections), but not deleted.

# SHOW COLLECTION

Table CD-2 shows the types of information that you can display in any combination about collections.

**Table CD-2: Types of Collection Display Information**

<b>Information Type</b>	<b>Qualifiers</b>
Amount of displayed information	<i>/BRIEF, /INTERMEDIATE, and /FULL</i>
Benchmark directory	<i>/BENCHMARK_DIRECTORY</i>
File output	<i>/OUTPUT</i>
CMS class	<i>/CLASS=BENCHMARK and /CLASS=TEMPLATE</i>
Template Directory	<i>/TEMPLATE_DIRECTORY</i>

---

## Command Qualifiers

### ***/BENCHMARK\_DIRECTORY***

Displays the directory specification of the benchmark directory used by this collection.

### ***/BRIEF***

Displays the collection name. The default is */INTERMEDIATE*.

### ***/CLASS=(keyword=class-name,...)***

Specifies the optional CMS class for benchmark files and template files stored in CMS libraries. The keywords, *BENCHMARK* and *TEMPLATE*, designate the name of the specific set of generations of elements. If you do not specify a class and the file is stored in a CMS library, the latest generation on the main line of descent is used. See the *Guide to VAX DEC/Code Management System* for more information about classes.

You can specify the same class names for your benchmark and template files. If you specify both keywords, separate them with a comma and enclose the list in parentheses. If you specify only one keyword, omit the parentheses.

# SHOW COLLECTION

## ***/FULL***

Displays the following output:

Collections in the DEC/Test Manager Library library-name

```
COLLECTION_NAME  NUMBER tests  DATE  TIME "remark"
  Command: COMMAND-LINE
  Status: RUN, COMPARISON, AND REVIEW STATUS
  Successful count: NUMBER      Unsuccessful count: NUMBER
  New test count: NUMBER        Updated test count: NUMBER
  Test not run count: NUMBER    Comparisons aborted: NUMBER
  Default template directory: DIRECTORY_SPECIFICATION
  Template class: CLASS_NAME
  Default benchmark directory: DIRECTORY_SPECIFICATION
  Benchmark class: CLASS_NAME
  Prologue: FILE_SPECIFICATION
  Epilogue: FILE_SPECIFICATION
  Last Review: DATE TIME
  Result Description COLLECTION_NAME      Comparison Status: Successful

  Benchmark file is device:[username]COLLECTION_NAME.BMK
  Result file does not exist
  Difference file does not exist
```

A result description for each test in the collection follows this display. A result description lists the test name and its comparison status and the file specifications for the benchmark, result, and difference files. The default is ***/INTERMEDIATE***.

## ***/INTERMEDIATE***

Displays the following output:

Collections in the DEC/Test Manager Library library-name

```
COLLECTION NAME  NUMBER tests  DATE  TIME "remark"
  Command: COMMAND-LINE
  Status: RUN, COMPARISON, AND REVIEW STATUS
  Successful count: NUMBER      Unsuccessful count: NUMBER
  New test count: NUMBER        Updated test count: NUMBER
  Test not run count: NUMBER    Comparisons aborted: NUMBER
```

## ***/OUTPUT[=file-specification]***

Sends the SHOW COLLECTION information to the specified file. If you specify the file name but omit the file type, the file type defaults to .LIS. The output is sent to SYS\$OUTPUT.

# SHOW COLLECTION

## ***/TEMPLATE\_DIRECTORY***

Displays the directory specification of the template directory used by this collection. Displays the directory specification of the default directory for the current library.

---

## **Example**

```
DTM> SHOW COLLECTION/BRIEF
```

```
Collections in DEC/Test Manager Library DUA0:[USER01.DTMLIB]
```

```
MAIL_COLL    MAIL_SEND_COLL    MAIL_SHOW_COLL
```

```
DTM>
```

**This command lists the names of all collections in the current library.**

# SHOW EPILOGUE

---

## SHOW EPILOGUE

Displays the file specification for the default epilogue file for collections.

---

### Format

**SHOW EPILOGUE** *[/qualifier]*

Command Qualifier	Default
/OUTPUT[=file-specification]	/OUTPUT=SYS\$OUTPUT

---

### Description

The SHOW EPILOGUE command displays the file specification for the default epilogue file for collections. The default collection epilogue file is established with the SET EPILOGUE command.

---

### Command Qualifier

***/OUTPUT[=file-specification]***

Sends the requested information to the specified file. If you specify the file name but omit the file type, the file type defaults to .LIS. The output is sent to SYS\$OUTPUT.

---

### Example

```
DTM> SHOW EPILOGUE
DUA0:[USER01.PROEPILIB]EPILOGUE_1.COM  "Default collection epilogue file
DTM>
```

This example displays the file specification for the default collection epilogue file.



---

# SHOW GROUP

Displays the contents of one or more groups.

---

## Format

**SHOW GROUP** *group-expression* [/qualifier...]

Command Qualifiers	Defaults
/BRIEF	/INTERMEDIATE
/[NO]CONTENTS[=n]	/CONTENTS=1
/FULL	/INTERMEDIATE
/INTERMEDIATE	/INTERMEDIATE
/MEMBER	None
/OUTPUT[=file-specification]	/OUTPUT=SYS\$OUTPUT

---

## Command Parameter

***group-expression***

Specifies the groups about which information is to be displayed. A group expression can be a group name or a list of group names separated by commas. You can use wildcards.

---

## Description

The SHOW GROUP command lists a group's name, its contents, and its creation remark. If you specify the SHOW GROUP command for more than one group, the groups are described in alphabetical order.

If you omit the group expression parameter, DEC/Test Manager displays information about all groups.

# SHOW GROUP

---

## Command Qualifiers

### ***/BRIEF***

Lists the group name only. You cannot specify both the ***/CONTENTS*** and ***/BRIEF*** qualifiers with the same command.

### ***/CONTENTS[=n]***

### ***/CONTENTS[=ALL]***

### ***/NOCONTENTS***

Displays the name of each group and the names of all test descriptions and groups that are contained in the specified groups. The default is ***/CONTENTS=1***.

The ***/CONTENTS=n*** qualifier displays this information for n levels of groups within each specified group. If n is greater than one, all nested groups for n levels are expanded and displayed.

The ***/CONTENTS=ALL*** qualifier displays this information for all groups within each group. The ***/NOCONTENTS*** qualifier displays the group names only.

### ***/FULL***

Displays the same output as the ***/CONTENTS=ALL*** qualifier.

### ***/INTERMEDIATE***

Lists the group name and any remark added during the creation, modification, or copying of the group. This is the default.

### ***/MEMBER***

Lists the groups of which the specified groups are members.

### ***/OUTPUT[=file-specification]***

Sends the requested information to the specified file. If you specify the file name but omit the file type, the file type defaults to ***.LIS***. The output is sent to ***SYS\$OUTPUT***.

---

## Example

```
DTM> SHOW GROUP/CONTENTS=ALL MARGINS
Groups in DEC/Test Manager Library DISK$USER01:[PROJECT.DTMLIB]

MARGINS "test of margin commands"
  LEFTMARGINS/GROUP
    LMTEST1
    LMTEST2
  RIGHTMARGINS/GROUP
    RMTEST1
    RMTEST2
  TEST1
```

This example displays the contents of the MARGINS group. Because /CONTENTS=ALL is specified, each group, test description, and remark is listed.

# SHOW HISTORY

---

## SHOW HISTORY

Displays a chronological list of the transactions performed on your DEC/Test Manager library.

---

### Format

**SHOW HISTORY** [*object-expression*] [*/qualifier...*]

**Command Qualifiers**

/BEFORE=time

/OUTPUT=file-specification

/SINCE=time

/[NO]TRANSACTIONS=(keyword,...)

/USER=user-name

**Defaults**

See text

/OUTPUT=SYS\$OUTPUT

Library creation

/TRANSACTIONS=ALL

None

---

### Command Parameter

***object-expression***

Specifies a DEC/Test Manager object about which history information is to be displayed. The object expression can be a test name, group name, collection name, variable name, or a list of these names separated by commas. You can use wildcards.

---

### Description

The SHOW HISTORY command displays a chronological list of transactions performed on a DEC/Test Manager library. If you specify an object expression, history information about that DEC/Test Manager entity is displayed. DEC/Test Manager records all transactions that alter the library

The SHOW HISTORY command qualifiers determine which library transactions should be reported. The qualifiers you use specify all the conditions that must be true for a particular transaction record to be printed.

# SHOW HISTORY

The following is an example transaction record:

```
12-MAY-1989 17:15:31 USER01 CREATE TEST MAIL_TEST "Going to test MAIL"
```

This record shows that on May 12, 1989 at 5:15 PM, USER01 issued the DEC/Test Manager CREATE TEST\_DESCRIPTION command to create the MAIL\_TEST test.

When you issue the DELETE HISTORY or REMARK command, DEC/Test Manager displays an asterisk (\*) in the first column of the transaction record.

Because transaction records for tests, collections, groups, or variables that are deleted from the library are retained, the SHOW HISTORY command can display records for tests and groups that do not currently exist. If you use a deleted name again, SHOW HISTORY does not distinguish between the old and new histories.

---

## Command Qualifiers

### ***/BEFORE=time***

Lists all history information prior to a specified date. The time is the current date and time, and the information displayed is whatever was logged prior to the issuance of the SHOW HISTORY command. The time value can be an absolute, delta, or combination time value, or one of the following keywords: TODAY, TOMORROW, or YESTERDAY.

### ***/OUTPUT=file-specification***

Sends the requested information to the specified file. If you specify the file name but omit the file type, the file type defaults to .LIS. The output is sent to SYS\$OUTPUT.

### ***/SINCE=time***

Specifies that only those history entries dated on or after the given time are to be displayed. All transactions recorded since the library was created are displayed. The time value can be an absolute, delta, or combination time value, or one of the following keywords: TODAY, TOMORROW, or YESTERDAY. If you specify the /SINCE qualifier but do not specify a value, DEC/Test Manager defaults to /SINCE=TODAY.

# SHOW HISTORY

***/TRANSACTION=(keyword,...)***

***/NOTTRANSACTION=(keyword,...)***

Displays all transaction records generated by the commands associated with the keywords you specify. The default is to list transaction records for all transactions in the DEC/Test Manager library. The valid keyword values are as follows:

ALL	Displays all transactions recorded in the history file
COPY	Displays all COPY commands
CREATE	Displays all CREATE commands
DELETE	Displays all DELETE commands
INSERT	Displays all INSERT commands
MODIFY	Displays all MODIFY commands
RECORD	Displays all RECORD commands
RECREATE	Displays all RECREATE commands
REMARK	Displays all REMARK commands
REMOVE	Displays all REMOVE commands
REVIEW	Displays all REVIEW commands
RUN	Displays all RUN commands
SET	Displays all SET commands
STOP	Displays all STOP commands
SUBMIT	Displays all SUBMIT commands
UPDATE	Displays all Review UPDATE commands
VERIFY	Displays all VERIFY commands

If you specify more than one command, you must enclose the list of commands in parentheses and separate them with commas.

The */TRANSACTION* qualifier directs DEC/Test Manager to list transaction records only for the listed commands.

The */NOTTRANSACTION* qualifier directs DEC/Test Manager to list transaction records for all commands except the listed commands.

***/USER=user-name***

Lists in chronological order the library transactions performed by the specified user. You cannot use wildcards in user names.

---

## Example

```
DTM> SHOW HISTORY
```

```
History in DEC/TEST MANAGER Library DUA0:[USER01.DTMLIB]
```

```
08-JAN-86 15:33:59 SMITH CREATE LIBRARY DUA1:[USER01.DTMLIB] "New  
DEC/Test Manager library"
```

```
08-JAN-86 15:55:45 SMITH CREATE TEST_DESCRIPTION RMTEST "Run MAIL test"
```

```
08-JAN-86 16:15:35 SMITH CREATE TEST_DESCRIPTION MEMO_TEST "First  
interactive test of memo template"
```

```
.
```

```
.
```

```
.
```

```
DTM>
```

**This command displays the history file for the current library.**

# SHOW LIBRARY

---

## SHOW LIBRARY

Displays the directory specification of the current DEC/Test Manager library.

---

### Format

**SHOW LIBRARY** *[/qualifier]*

Command Qualifier	Default
/OUTPUT[= <i>file-specification</i> ]	/OUTPUT=SYS\$OUTPUT

---

### Description

The SHOW LIBRARY command displays the file specification of the current DEC/Test Manager library, or informs you if the library is invalid or if no library has been set. If the library is invalid, DEC/Test Manager issues an error message instructing you to use the VERIFY command.

---

### Command Qualifier

***/OUTPUT[=*file-specification*]***

Sends the requested information to the specified file. If you specify the file name but omit the file type, the file type defaults to .LIS. The output is sent to SYS\$OUTPUT.

---

### Example

```
DTM> SHOW LIBRARY
Your DEC/Test Manager library is DUA0:[USER01.DTMLIB]
```

This example displays your current DEC/Test Manager library.



---

# SHOW PROLOGUE

Displays the file specification of the default prologue file for collections.

---

## Format

**SHOW PROLOGUE** *[/qualifier]*

Command Qualifier	Default
/OUTPUT[=file-specification]	/OUTPUT=SYS\$OUTPUT

---

## Description

The SHOW PROLOGUE command displays the file specification for the default prologue file for collections. This file specification was established with a previous SET PROLOGUE command.

---

## Command Qualifier

***/OUTPUT[=file-specification]***

Sends the requested information to the specified file. If you specify the file name but omit the file type, the file type defaults to .LIS. The output is sent to SYS\$OUTPUT.

---

## Example

```
DTM> SHOW PROLOGUE
DUA0:[USER01.PROEPILIB]PROLOGUE_1.COM "Default collection prologue file"
DTM>
```

This example displays the file specification for the default collection prologue file.

# SHOW TEMPLATE\_DIRECTORY

---

## SHOW TEMPLATE\_DIRECTORY

Displays the directory specification for the default directory that contains the template files.

---

### Format

**SHOW TEMPLATE\_DIRECTORY** *[/qualifier]*

**Command Qualifier**

*/OUTPUT[=file-specification]*

**Default**

*/OUTPUT=SYS\$OUTPUT*

---

### Description

The **SHOW TEMPLATE\_DIRECTORY** command displays the directory specification for the default directory that contains the template files. This directory can be a CMS library.

---

### Command Qualifier

***/OUTPUT[=file-specification]***

Sends the requested information to a specified file. If you specify the file name but omit the file type, the file type defaults to .LIS. The output is sent to SYS\$OUTPUT.

---

### Example

```
DTM> SHOW TEMPLATE_DIRECTORY
      DUA0:[USER01.TEMPLATE]                "Default template library"
DTM>
```

This example displays the directory specification for the current default template directory.

---

## SHOW TEST\_DESCRIPTION

Displays the files and groups associated with one or more test descriptions.

---

### Format

**SHOW TEST\_DESCRIPTION** [*test-group-expression*] [*/qualifier...*]

<b>Command Qualifiers</b>	<b>Defaults</b>
/BENCHMARK	See text
/BRIEF	/INTERMEDIATE
/COMMAND	See text
/COMPARISON_TYPE	See text
/EPILOGUE	See text
/FILTER	See text
/FULL	/INTERMEDIATE
/GROUPS	See text
/INTERMEDIATE	/INTERMEDIATE
/OUTPUT[=file-specification]	/OUTPUT=SYS\$OUTPUT
/PROLOGUE	See text
/REMARK	See text
/TEMPLATE	See text
/TYPE	See text
/VARIABLE	See text
<b>Parameter Qualifiers</b>	<b>Defaults</b>
/GROUP	/TEST_DESCRIPTION
/TEST_DESCRIPTION	/TEST_DESCRIPTION

---

### Restrictions

- The /COMMAND qualifier applies to DECwindows tests only.
- The /FILTER qualifier applies to noninteractive and terminal tests only.

# SHOW TEST\_DESCRIPTION

---

## Command Parameter

### *test-group-expression*

Specifies tests or groups of tests about which to display information. A test expression can be a test name or a group name or a list of group names separated by commas. You can use wildcards.

Use the /GROUP or /TEST\_DESCRIPTION qualifier to identify items in a test expression as part of a test or group.

---

## Description

The SHOW TEST\_DESCRIPTION command displays files and groups associated with the specified test descriptions. The following table describes the types of information that is displayed with the SHOW TEST\_DESCRIPTION command:

---

<b>Command Qualifier</b>	<b>Function</b>
/BENCHMARK /COMPARISON_TYPE /FILTER /EPILOGUE /PROLOGUE /TEMPLATE /VARIABLE /COMMAND	Displays the contents of the specified test description fields.
/GROUPS /REMARK /TYPE	Displays the groups and remark associated with the test description, and whether the test description describes an interactive or noninteractive test.
/BRIEF /INTERMEDIATE /FULL	Determines the amount of information displayed
/OUTPUT	Redirects the output from the command.

---

If you do not supply a test expression, DEC/Test Manager displays the intermediate level of information for all tests in the library.

---

## Command Qualifiers

### ***/BENCHMARK***

Displays the file specification for the benchmark file associated with the specified test description.

If you use the default benchmark when you create the test description, the benchmark file specification is displayed in the following format:

```
test-name.BMK
```

### ***/BRIEF***

Displays the test name.

### ***/COMMAND***

Displays the command that is used to begin recording and executing of the test. This qualifier applies to DECwindows tests only.

### ***/COMPARISON\_TYPE***

Displays the comparison type associated with the specified test description.

### ***/EPILOGUE***

Displays the file specification for the epilogue file associated with the specified test description.

### ***/FILTER***

Displays the names for all filters associated with the specified test description.

### ***/FULL***

Displays the contents of all test description fields. The format is as follows:

```
Test descriptions in DEC/Test Manager library library-name
```

Test_name	"remark"
Template	= file-name
Benchmark	= file-name
Prologue	= file-name
Epilogue	= file-name
Variables	= variable-name(s) [=value]
Groups	= group-name(s)
Type	= type-name
Command	= command
Filters	= filter-type(s)
Comparison Type	= comparison-type

# SHOW TEST\_DESCRIPTION

## ***/GROUPS***

Displays the names of the groups with which the test is affiliated. Because of the existence of the */GROUP* parameter qualifier for this command, you cannot abbreviate the */GROUPS* qualifier.

## ***/INTERMEDIATE***

Displays the contents of the benchmark, template, prologue, and epilogue test description fields. The format is as follows:

Test descriptions in DEC/Test Manager library library-name

```
Test_name      "remark"  
  Template =   file-name  
  Benchmark=   file-name  
  Prologue =   file-name  
  Epilogue =   file-name
```

This is the default.

## ***/OUTPUT[=file-specification]***

Sends the requested information to a specified file. If you specify the file name but omit the file type, the file type defaults to .LIS. By default, the output is sent to SYS\$OUTPUT.

## ***/PROLOGUE***

Displays the file specification for the prologue file associated with the specified test description.

## ***/REMARK***

Displays the remark associated with the test description.

## ***/TEMPLATE***

Displays the file specification for the template associated with the specified test description.

## ***/TYPE***

Displays the type of the specified test description—NONINTERACTIVE, INTERACTIVE, or DECWINDOWS.

## ***/VARIABLE***

Displays the names and values for all variables associated with the test description.

---

## Parameter Qualifiers

### ***/GROUP***

Identifies the immediately preceding item in the parameter as a group expression. If a test group expression comprises a list, use this qualifier after each item that designates a group.

Because of the existence of the /GROUPS qualifier for this command, you cannot abbreviate the /GROUP qualifier.

### ***/TEST\_DESCRIPTION***

Identifies the immediately preceding item in the parameter as a test expression. This is the default.

---

## Example

```
DTM> SHOW TEST DESCRIPTION SEND MAIL TEST
Test descriptions in DEC/Test Manager Library DUA0:[USER01.DTMLIB]

SEND_MAIL_TEST          "MAIL SEND command test"

    Template = SEND_MAIL_TEST.COM
    Benchmark= SEND_MAIL_TEST.BMK
    Prologue  = None Specified
    Epilogue  = None Specified
```

This example displays the contents of the test description SEND\_MAIL\_TEST.

# SHOW VARIABLE

---

## SHOW VARIABLE

Displays information about the specified variables.

---

### Format

**SHOW VARIABLE** *variable-expression* [/qualifier...]

<b>Command Qualifiers</b>	<b>Defaults</b>
/BRIEF	/INTERMEDIATE
/FULL	/INTERMEDIATE
/INTERMEDIATE	/INTERMEDIATE
/OUTPUT[=file-specification]	/OUTPUT=SYS\$OUTPUT
/REMARK	See text
/SCOPE	See text
/TEST_DESCRIPTION	See text
/TYPE	See text
/USAGE	See text
/VALUE	See text

---

### Command Parameter

***variable-expression***

Specifies the variable about which to display information. A variable expression can be a variable name or a list of variable names separated by commas. You can use wildcards.

---

### Description

The SHOW VARIABLE command displays information about the specified variables. The variable must have been previously defined in the DEC/Test Manager library with the CREATE VARIABLE command.

The default qualifier, /INTERMEDIATE, displays the name, value, and remark for a variable.



---

## Command Qualifiers

***/BRIEF***

Displays the variable name only.

***/FULL***

Displays the name, value, scope, usage, and remark for the variable and lists the tests with which the variable is associated.

***/INTERMEDIATE***

Displays the name, value, and remark for the variable. This is the default.

***/OUTPUT[=file-specification]***

Sends the requested information to the specified file. If you specify the file name but omit the file type, the file type defaults to .LIS. The output is sent to SYS\$OUTPUT.

***/REMARK***

Displays the remark associated with the variable.

***/SCOPE***

Displays the scope of the variable. The scope can be either global or local.

***/TEST\_DESCRIPTION***

Lists the test descriptions with which the variable is associated.

***/TYPE***

Displays the type of variable—STRING or NUMERIC.

***/USAGE***

Displays the use of the variable. The use can be either as a symbol or as a logical.

***/VALUE***

Displays the variable's default value. If no value has been supplied, the null value is displayed.

# SHOW VARIABLE

---

## Example

```
DTM> SHOW VARIABLE /VALUE INPUT_FILE
```

```
Variables in the DEC/Test Manager library DUA0:[USER01.PROJECT]
```

```
INPUT_FILE      "input file for DSR tests"  
Value = INPUT.RNO
```

**This example displays the value of the variable INPUT\_FILE.**

---

## SHOW VERSION

Displays the version number for DEC/Test Manager currently in use on the system.

---

### Format

**SHOW VERSION** *[/qualifier]*

**Command Qualifier**

*/OUTPUT[=file-specification]*

**Default**

*/OUTPUT=SYS\$OUTPUT*

---

### Description

The SHOW VERSION command shows you what version of DEC/Test Manager you are using. You should include the full text of this message when you submit Software Performance Reports (SPR).

---

### Command Qualifier

***/OUTPUT[=file-specification]***

Sends requested information to the specified file. If you specify the file name but omit the file type, the file type defaults to .LIS. The output is sent to SYS\$OUTPUT.

---

### Example

```
DTM> SHOW VERSION
DEC/Test Manager Version 3.1
```

This example displays the version number of DEC/Test Manager.

# SPAWN

---

## SPAWN

Creates a subprocess from the current DEC/Test Manager session—the parent process.

---

### Format

**SPAWN** [*command*] [*/qualifier...*]

<b>Command Qualifiers</b>	<b>Defaults</b>
/[NO]CARRIAGE_CONTROL	See text
/[NO]CLI[=cli]	See text
/INPUT=file-specification	See text
/[NO]KEYPAD	/KEYPAD
/[NO]LOGICAL_NAMES	/LOGICAL_NAMES
/[NO]NOTIFY	/NONOTIFY
/OUTPUT=file-specification	See text
/PROCESS=subprocess-name	See text
/[NO]PROMPT[=string]	See text
/[NO]SYMBOLS	/SYMBOLS
/[NO]WAIT	/WAIT

---

### Command Parameter

#### ***command***

Specifies an optional command to be executed by the subprocess you are creating. If you specify the command parameter, you create a subprocess that executes the command and returns control to the DEC/Test Manager session when the command terminates.

If you include the /INPUT qualifier with the command parameter, the subprocess reads commands from the specified input file after the command executes.

The command string can consist of up to 132 characters.

If you omit the command parameter, the SPAWN command creates a subprocess and attaches your terminal to it. You can return to your DEC/Test Manager session by logging out of the subprocess or by issuing the ATTACH command with the /PARENT qualifier. If you have created several subprocesses, you can switch between them using the ATTACH command with the /IDENTIFICATION qualifier.

---

## **Description**

The SPAWN command creates a subprocess from the current DEC/Test Manager session—the parent process. The context of your DEC/Test Manager session is copied to the subprocess.

You can use the SPAWN command to leave DEC/Test Manager temporarily, to create another DEC/Test Manager session, or to edit a file and then return to your original DEC/Test Manager session.

---

## **Command Qualifiers**

### ***/CARRIAGE\_CONTROL***

### ***/NOCARRIAGE\_CONTROL***

Determines whether carriage control characters or line feed characters, or both are prefixed to the DCL-prompt string of the subprocess. The default is the current setting of the parent process.

### ***/CLI[=*cli*]***

### ***/NOCLI***

Specifies an alternate command language interpreter (CLI) for the subprocess to use. The default is the CLI the parent process uses.

The CLI you specify must be located in SYS\$SYSTEM and have the file type .EXE.

### ***/INPUT=file-specification***

Specifies an input file containing one or more commands for the spawned subprocess to execute. If you specify a command with an input file, the command is processed before the commands in the input file. The subprocess terminates when processing is complete. You cannot use wildcards in the file specification.

# SPAWN

## ***/KEYPAD (D)***

### ***/NOKEYPAD***

Determines whether DCL keypad symbols and the current DCL keypad state are copied from the parent process to the subprocess. Use the ***/NOKEYPAD*** qualifier if you do not want the key settings to be copied.

## ***/LOGICAL\_NAMES (D)***

### ***/NOLOGICAL\_NAMES***

Determines whether the system passes process logical names and logical name tables to the subprocess, except those marked **CONFINE** or those created in executive or kernel mode.

## ***/NOTIFY***

### ***/NONOTIFY (D)***

Determines whether a message is sent to your terminal to notify you that your subprocess has been completed or aborted. You must specify the ***/NOWAIT*** qualifier with the ***/NOTIFY*** qualifier.

## ***/OUTPUT=file-specification***

Specifies the output file to which the output of the SPAWN operation is to be written. If you specify the ***/NOWAIT*** qualifier, use the ***/OUTPUT*** qualifier to specify an output other than **SYS\$OUTPUT** to prevent your terminal from being used by both processes simultaneously. The default is to direct output to the current **SYS\$OUTPUT** device.

## ***/PROCESS=subprocess-name***

Specifies the name of the subprocess to be created. The default name for the subprocess is **USERNAME\_n** (where **n** denotes a unique number).

## ***/PROMPT[=string]***

Specifies the prompt string for the subprocess. If you specify the ***/PROMPT*** qualifier but do not specify a string, the default prompt is displayed. The default is to copy the current prompt string from the parent process.

## ***/SYMBOLS (D)***

### ***/NOSYMBOLS***

Determines whether the system passes DCL global and local symbols to the subprocess.

***/WAIT (D)***  
***/NOWAIT***

Controls whether the system waits until the subprocess is completed before allowing more commands to be issued by the parent process. The */NOWAIT* qualifier enables you to enter more commands while the specified subprocess is running. If you specify the */NOWAIT* qualifier, you should also specify */OUTPUT* to direct output to a file (rather than to the screen). This prevents your terminal from being used by both processes simultaneously.

---

## Example

```
DTM> SPAWN MAIL
```

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

```
MAIL>
```

This example spawns the VMS MAIL Utility from the DEC/Test Manager subsystem. Enter the *ATTACH* command to terminate the MAIL session and return to the DEC/Test Manager system level.

# STOP

---

## STOP

Stops a collection that has been submitted to a batch queue.

---

### Format

**STOP** *collection-name* [/qualifier...] "remark"

Command Qualifiers	Defaults
/[NO]CONFIRM	/NOCONFIRM
/[NO]LOG	/LOG

---

### Command Parameters

***collection-name***

Specifies the collection that is to be stopped. You cannot use wildcards to specify the collection name parameter.

***remark***

Specifies a string that contains a comment. You must specify a remark within quotation marks (" "). The exception to this rule is when you specify a remark string at the remark prompt. If you do not provide a remark string, you are prompted for one. However, a null remark is permitted.

---

### Description

The STOP command stops the execution of a collection of tests that has been submitted to a batch queue. The batch job stops and is removed from the batch queue, necessary clean up is performed on the database, and you see a message. All tests that have completed are available for comparison and review.

Pressing CTRL/C stops a collection running interactively.



You must have the appropriate privileges to stop a collection submitted by someone else. If the person is in your UIC group, you must have GROUP privilege. If the person is not in your UIC group, you must have WORLD privilege.

---

## Command Qualifiers

***/CONFIRM***

***/NOCONFIRM (D)***

Controls whether DEC/Test Manager prompts you to confirm that you want to stop the collection.

***/LOG (D)***

***/NOLOG***

Controls whether DEC/Test Manager displays informational and success messages on your screen.

---

## Example

```
DTM> STOP MAIL_COLL/CONFIRM "Stopping run of MAIL_COLL"  
Confirm stop of collection MAIL_COLL [Y/N] (N): Y  
%DTM-S-COLSTOPPED, collection MAIL_COLL has been stopped  
DTM>
```

This example stops collection MAIL\_COLL that is executing in batch.

# SUBMIT

---

## SUBMIT

Executes a collection in batch producing result files that are subsequently compared with benchmark files.

---

### Format

**SUBMIT** *collection-name* [/qualifier...] "remark"

Command Qualifiers	Defaults
/[NO]CONFIRM	/CONFIRM
/[NO]KEEP	/KEEP
/[NO]LOG	/LOG

---

### Restriction

- You can submit collections that contain DECwindows tests but you must be logged into the DECwindows environment and initialize it for testing

---

### Command Parameters

***collection-name***

Specifies the collection to execute in batch mode. You cannot use wildcards to specify the collection name parameter.

***remark***

Specifies a string that contains a comment. You must specify a remark within quotation marks (" "). The exception to this rule is when you specify a remark string at the remark prompt. If you do not provide a remark string, you are prompted for one. However, a null remark is permitted.

---

## Description

The **SUBMIT** command executes a collection of tests in batch. It processes test template files or previously recorded session files and produces a result file for each test in the collection. The result files are subsequently compared to their respective benchmark files.

You can use the following DCL **SUBMIT** command qualifiers with the **SUBMIT** command:

<b>/AFTER</b>	<b>/CHARACTERISTICS</b>	<b>/CPUTIME</b>
<b>/[NO]HOLD</b>	<b>/[NO]LOG_FILE</b>	<b>/NAME</b>
<b>/[NO]NOTIFY</b>	<b>/[NO]PRINTER</b>	<b>/PRIORITY</b>
<b>/QUEUE</b>	<b>/USER</b>	<b>/WSDEFAULT</b>
<b>/WSEXTENT</b>	<b>/WSQUOTA</b>	

See the *VMS DCL Dictionary* for more information about these qualifiers.

You can run collections more than once. When you rerun a collection of tests with the **SUBMIT** command, the collection is not re-created. The original collection is run again. For example, if you have changed any of the tests or the contents of any groups contained in the collection, these changes are not reflected in the resubmitted test run. If you issue the **SUBMIT** command to rerun a collection that you have not reviewed, DEC/Test Manager prompts you for confirmation before rerunning the collection.

Use the **RECREATE** command to re-create a collection with the most current version of all the files in the collection.

Note that you cannot use the DCL **SUBMIT** command to submit a collection. You must use the DEC/Test Manager **SUBMIT** command.

---

## Command Qualifiers

**/CONFIRM**  
**/NOCONFIRM (D)**

Controls whether DEC/Test Manager prompts you to confirm that you want to submit the collection again if it has not been reviewed. If you have executed but not reviewed a collection, you are prompted to confirm resubmission of the collection.

# SUBMIT

***/KEEP (D)***

***/NOKEEP***

Controls whether the log file is deleted after it is printed.

***/LOG (D)***

***/NOLOG***

Controls whether DEC/Test Manager displays informational and success messages on your screen.

---

## Example

```
DTM> SUBMIT MAIL_COLL/NOTIFY/LOG_FILE=[]/QUEUE=SYS$LARGE
%DTM-S-SUBMITTED, collection MAIL_COLL submitted
-DTM-I-TEXT, Job MAIL_COLL (queue SYS$LARGE entry 1000) started on
SYS$LARGE
```

This example submits the collection MAIL\_COLL to the batch queue SYS\$LARGE. The log file is left in the default directory and you are notified when the collection is finished running.

---

## VERIFY

Evaluates the integrity of the current DEC/Test Manager library and its collections.

---

### Format

**VERIFY** *[/qualifier...]* "remark"

Command Qualifiers	Defaults
/[NO]LOG	/LOG
/RECOVER	None
/[NO]REPAIR	/NOREPAIR

---

### Command Parameter

***remark***

Specifies a string that contains a comment. You must specify a remark within quotation marks (" "). The exception to this rule is when you specify a remark string at the remark prompt. If you do not provide a remark string, you are prompted for one. However, a null remark is permitted.

---

### Description

When you issue the VERIFY command, DEC/Test Manager performs a series of evaluations on the current DEC/Test Manager library and its collections to confirm that the library structure, the collections, and the library files are in a valid form.

If the library and collections are valid, the command executes successfully. If the library or the collections are invalid, DEC/Test Manager informs you to use the /RECOVER qualifier to correct some of the errors the VERIFY command discovers.

# VERIFY

When you specify the VERIFY command with the /RECOVER qualifier and DEC/Test Manager encounters a subdirectory of the DEC/Test Manager library that contains files but is not associated with a collection, DEC/Test Manager issues a confirmation message before deleting the directory. See Chapter 7 for more information on recovering an invalid DEC/Test Manager library.

---

## Command Qualifiers

### ***/LOG (D)***

### ***/NOLOG***

Controls whether DEC/Test Manager displays informational and success messages on your screen.

### ***/RECOVER***

Attempts to restore the current library and its collections to a usable state in the event of a process or job abortion or a system failure. It also cleans up the library by deleting files and directories that DEC/Test Manager does not own.

### ***/REPAIR***

### ***/NOREPAIR (D)***

Attempts to reclaim all loose blocks in the current DEC/Test Manager library and to delete illegal files found in the library. If DEC/Test Manager encounters a subdirectory of the DEC/Test Manager library that contains files and the directory is not associated with a collection, DEC/Test Manager issues a confirmation message before deleting the directory.

---

## Examples

```

1. DTM> VERIFY
%DTM-I-VERFRE, free space list verified
%DTM-I-VERSTR, string list verified
%DTM-I-VERCOL, collection list verified
%DTM-I-VERGRO, group list verified
%DTM-I-VERTD, test description list verified
%DTM-I-VERVAR, variables list verified
%DTM-I-VERARC, archive list verified
%DTM-I-VERHEAD, user header information verified
%DTM-I-VERSPACE, contiguous space verified
%DTM-I-VERCOLDIR, collection directory structure verified
%DTM-S-VERIFIED, DEC/Test Manager library DUA0:[USER01.DTMLIB]
verified
DTM>

```

This example verifies that the current DEC/Test Manager library is valid.

```

2. DTM> VERIFY/RECOVER
%DTM-S-RECNOTNEC, recovery is not necessary; DEC/Test Manager library
DUA0:[USER01.DTMLIB] is in a safe state
DTM>

```

This example verifies that the current DEC/Test Manager library and its collections are valid and informs you that you do not need to recover the library or its collections.

---

## 4 Review Subsystem Command Descriptions

The Review Subsystem commands are arranged in alphabetical order with each command description containing the following:

- Command Format
- Restrictions, if any
- Command parameters
- Descriptions of the command
- Command qualifiers (defaults, if any, are marked (D))
- Parameter qualifiers, if any (defaults, if any, are marked (D))
- Examples

## DTM\_REVIEW> @file-specification

---

## DTM\_REVIEW> @file-specification

Executes DEC/Test Manager Review subsystem commands contained in the specified file.

---

### Format

**@file-specification**

**Command Qualifiers**

None

**Defaults**

None

---

### Restriction

- You can only specify Review subsystem commands with this @file-specification command.

---

### Command Parameter

***file-specification***

Specifies the command procedure you want to execute. If the file specification does not include a file type, DEC/Test Manager assumes the default file type .COM.

---

### Description

The @file-specification command executes the Review subsystem commands in the specified file. The file can contain any DEC/Test Manager Review subsystem command, including another @file-specification command. Do not preface the Review subsystem commands in a file with DTM or \$. For example, enter BACK 5 not DTM BACK 5.



## DTM\_REVIEW> @file-specification

When DEC/Test Manager executes an EXIT command or reaches the end of the command procedure, it leaves you at the current system level. The invoking command stream can be either the terminal or another command procedure.

---

### Example

```
DTM_REVIEW> @REVIEW_FILE
Result Description MAIL_TEST      Comparison Status : Unsuccessful
Differences File DUA0:[USER01.DTMLIB]MAIL_TEST.DIF For Result
Description MAIL_TEST

DEC/Test Manager COMPARE utility
(1) DUA0:[USER01.DTMLIB]MAIL_TEST.BMK
(2) DUA0:[USER01.DTMLIB.RUNMAIL]MAIL_TEST.RES
.
.
.
```

This example executes the command procedure REVIEW\_FILE, which contains the NEXT/UNSUCCESSFUL and SHOW/DIFFERENCES commands.

## DTM\_REVIEW> ATTACH

---

## DTM\_REVIEW> ATTACH

Switches control from the current process to another process in your job.

---

### Format

**ATTACH** *process-name* [/qualifier...]

Command Qualifiers	Defaults
/IDENTIFICATION=pid	None
/PARENT	None

---

### Command Parameter

***process-name***

Specifies an existing process to which you want to attach the terminal.

If you specify either the /IDENTIFICATION or /PARENT qualifier, do not specify the process name parameter or the other qualifier. If you do not specify a qualifier, you must specify a process name.

---

### Description

The ATTACH command enables you to connect the terminal to another process in your job. You can use the ATTACH command to change control between subprocesses you have created with the SPAWN command or to reconnect to the parent process. You can also use the ATTACH command in conjunction with the SPAWN/WAIT command to return to the Review session without terminating the subprocess. See the SPAWN command for more information.

---

## Command Qualifiers

### ***/IDENTIFICATION=pid***

Specifies the process identification (PID) of the process to which you want to attach the terminal. You can omit the leading zeros when you specify a PID.

If you specify the */IDENTIFICATION* qualifier, do not specify the process name parameter or the */PARENT* qualifier. If you do not specify a qualifier, you must specify a process name.

### ***/PARENT***

Specifies that the process you want to attach to is the original (parent) process.

If you specify the */PARENT* qualifier, do not specify the process name parameter or the */IDENTIFICATION* qualifier. If you do not specify a qualifier, you must specify a process name.

---

## Example

```
MAIL> SPAWN DTM REVIEW
Collection name: MAIL_COLL
Collection MAIL_COLL with 1 test was created on 29-JUL-85 15:13:54
by the command:
    CREATE COLLECTION MAIL_COLL MAIL_TEST "Recording MAIL on the terminal"
    Last review status = not previously reviewed
    Success count = 0
    Unsuccessful count = 1
    New test count = 0
    Updated test count = 0
    Comparisons aborted = 0
    Test not run count = 0

Result Description MAIL_TEST      Comparison Status : Unsuccessful

DTM_REVIEW> ATTACH/PARENT
You have 0 new messages.
MAIL>
```

This example uses the MAIL SPAWN command to create a subprocess executing a DEC/Test Manager Review subsystem to review collection MAIL\_COLL. The Review subsystem ATTACH command is then used to attach the terminal back to the mail session, the parent process.

## DTM\_REVIEW> BACK

---

## DTM\_REVIEW> BACK

Moves you backward through the sequence of result descriptions being reviewed.

---

### Format

**BACK** [*count*] [/*qualifier...*]

#### Command Qualifiers

/COMPARISON\_ABORTED  
/NEW  
/NOT\_RUN  
/SUCCESSFUL  
/UNSUCCESSFUL  
/UPDATED

#### Defaults

Previous result description  
Previous result description  
Previous result description  
Previous result description  
Previous result description  
Previous result description

---

### Command Parameter

#### *count*

An integer that indicates the number of result descriptions to move backward from the current result description. The default is 1.

---

### Description

The BACK command moves you backward through the sequence of result descriptions being reviewed. The optional count parameter is the number of result descriptions to move backward from the current result description. If the parameter is omitted, you move back to the previous result description. The result description to which you move backward becomes the current result description.

The comparison status qualifiers */COMPARISON\_ABORTED*, */NEW*, */NOT\_RUN*, */SUCCESSFUL*, */UNSUCCESSFUL*, and */UPDATED* move you back to the most recent result description with the specified comparison status. Specifying a count parameter in combination with a comparison status qualifier moves you back through the specified number of result descriptions with the specified comparison status.

---

## Command Qualifiers

### */COMPARISON\_ABORTED*

Moves you to the previous result description for a test whose comparison aborted.

### */NEW*

Moves you to the previous result description for a new test.

### */NOT\_RUN*

Moves you to the previous result description for a test that did not run.

### */SUCCESSFUL*

Moves you to the previous successfully compared result description.

### */UNSUCCESSFUL*

Moves you to the previous unsuccessfully compared result description.

### */UPDATED*

Moves you to the previous updated result description.

---

## Example

```
DTM_REVIEW> BACK 2
.
```

This example displays the result description that is two positions behind the current result description.

## DTM\_REVIEW> DEFINE/KEY

---

## DTM\_REVIEW> DEFINE/KEY

Defines a key to execute a command string.

---

### Format

**DEFINE/KEY** *key-name* "command-string" [/qualifier...]

#### Command Qualifiers

/[NO]ECHO  
/[NO]IF\_STATE=(state-name,...)  
/[NO]LOCK\_STATE  
/[NO]SET\_STATE=state-name  
/[NO]TERMINATE

#### Defaults

/ECHO  
Current state  
/NOLOCK\_STATE  
Current state  
/NOTERMINATE

---

### Command Parameters

#### *key-name*

Specifies the key to define. You can use the DEFINE\KEY command to define the following keys:

PF1 to PF4  
KP0 to KP9  
period (.)  
comma (,)  
minus (-)  
Enter  
left arrow (←)  
right arrow (→)  
Find (E1)  
Insert Here (E2)  
Remove (E3)  
Select (E4)  
Prev Screen (E5)  
Next Screen (E6)  
Help

Do  
F6 to F20

***command-string***

Specifies the command string to be entered when you press the defined key. The command string can be a Review subsystem command. If the command contains any spaces, enclose the command string in quotation marks (" ").

---

## Description

The DEFINE/KEY command defines a key to issue a Review subsystem command. You can customize your keyboard by defining keys to issue often used commands or command strings that are long.

The definitions you create with the DEFINE/KEY command are in effect only for the current Review subsystem session; the next time you invoke the Review subsystem, only the default key definitions will be in effect. To save your key definitions for use in every Review subsystem session, include them in an initialization file. This file is executed whenever you invoke the Review subsystem as a subsystem. See Section 6.6.2 for more information on the initialization file.

If you have key definitions that you want to save but do not necessarily want to use every time you invoke the Review subsystem, store them in a command procedure. See the *Guide to Using DCL and Command Procedures on VMS* for more information on DEC/Test Manager command procedures.

DEC/Test Manager provides a set of default definitions. You can use the DEFINE/KEY command to replace these definitions or to define certain undefined keys. Pressing the PF2 key displays the default key definitions.

The state name value used with the /IF\_STATE, /LOCK\_STATE, and /SET\_STATE qualifiers can be any alphanumeric string. The state names defined by DEC/Test Manager are DTM and GOLD\_DTM.

You can define the GOLD key (PF1) to execute DEC/Test Manager commands in two keystrokes by using the DEFINE/KEY command with the /SET\_STATE=GOLD\_DTM qualifier. By doing this, you can provide two definitions to the same key. For example, you can define KP1 to issue the CREATE GROUP command and define GOLD-KP1 to issue the MODIFY GROUP command.

---

## Command Qualifiers

***/ECHO (D)***

***/NOECHO***

Specifies whether the command is displayed on your screen after you press the defined key. You cannot define a key by specifying both */NOECHO* and */NOTERMINATE*.

***/IF\_STATE=(state-name,...)***

***/NOIF\_STATE***

Specifies a list of states, any one of which must be set to enable the specific key definition. The default is the current state. DEC/Test Manager defines the two states *REVIEW* and *GOLD\_REVIEW*. The */NOIF\_STATE* qualifier selects the current state.

***/LOCK\_STATE***

***/NOLOCK\_STATE (D)***

Retains the state specified with the */SET\_STATE* qualifier until you use the */SET\_STATE* qualifier again to change it.

***/SET\_STATE=state-name***

***/NOSET\_STATE***

Associates a state with the key you are defining. The default is the current state. A state name can be any alphanumeric string. DEC/Test Manager defines the two states *REVIEW* and *GOLD\_REVIEW*. You cannot define a key specifying both */SET\_STATE* and */TERMINATE*. The */NOSET\_STATE* qualifier selects the current state.

***/TERMINATE***

***/NOTERMINATE (D)***

Determines whether the specified command string executes when you press the defined key. When you use */NOTERMINATE*, you must press *RETURN* to execute the command. You cannot define a key specifying both */SET\_STATE* and */TERMINATE* or */NOECHO* and */NOTERMINATE*.



---

## Example

```
DTM_REVIEW> DEFINE/KEY KP1/IF_STATE=REVIEW/TERMINATE-  
_DTM_REVIEW> "NEXT/SUCCESSFUL"  
DTM_REVIEW> DEFINE/KEY KP1/IF_STATE=GOLD_REVIEW/TERMINATE-  
_DTM_REVIEW> "BACK/SUCCESSFUL"  
DTM_REVIEW>
```

This example defines two keys on the Review subsystem keypad. When you press the keypad 1 key, you move forward to the next result description for a successful test. When you press the GOLD key followed by the keypad 1 key, you move backward to the previous result description for a successful test.

## DTM\_REVIEW> EXIT

---

## DTM\_REVIEW> EXIT

Terminates the Review session and returns control to the previous command level where you invoked the Review subsystem.

---

### Format

**EXIT** [*/qualifier...*]

Command Qualifiers	Defaults
/NOINSERT	Inserts test descriptions into group
/NOPRINT	Prints files

---

### Description

The EXIT command terminates the Review session and returns control to the previous command level where you invoked the Review subsystem. Unless you specify the /NOINSERT or /NOPRINT qualifier, tests marked with the INSERT command are placed in a group and files selected with the PRINT command are placed in the system's default printer queue.

You can also press CTRL/Z to leave the Review subsystem as if you entered the EXIT command. You can press CTRL/C to leave the Review subsystem, but it is as if you entered the EXIT command with the /NOINSERT and /NOPRINT qualifiers.

---

### Command Qualifiers

#### ***/NOINSERT***

Specifies that a group is not to be created from tests marked with the INSERT command when you exit the Review subsystem.

#### ***/NOPRINT***

Specifies that files marked with the PRINT command are not to be printed when you exit the Review subsystem.

---

## **Example**

```
DTM_REVIEW> EXIT
%DTM-S-EXIT, leaving Review subsystem
DTM>
```

**This example shows how to exit from the Review subsystem.**

## DTM\_REVIEW> FIRST

---

# DTM\_REVIEW> FIRST

Moves you to the first result description in the collection you are reviewing

---

## Format

### FIRST

Command Qualifiers	Defaults
--------------------	----------

None

None

---

## Description

The **FIRST** command moves you to the first result description in the collection you are reviewing. The first result description becomes the current result description.

---

## Example

```
DTM_REVIEW> FIRST
```

```
.  
.  
.
```

This example displays the first result description in a collection.

---

## DTM\_REVIEW> HELP

Displays help text for Review subsystem commands.

---

### Format

**HELP** *[topic]*

**Command Qualifiers**

None

**Defaults**

None

---

### Command Parameter

***topic***

Specifies a DEC/Test Manager Review subsystem subject about which you want information. A topic can be either a subject (such as **KEYPAD**) that is discussed in the Review subsystem help file, or a Review subsystem command. (The command can include qualifiers.) A list of Review subsystem help topics appears after you issue the **HELP** command.

---

### Description

The **HELP** command displays Review subsystem information on the screen.

The optional topic parameter enables you to get help on specific topics and on all Review subsystem commands. If you do not specify a topic, you get a display of available **HELP** features and instructions for displaying the **HELP** text. If you specify a topic, information is displayed about that topic. If you specify a Review subsystem command, information is displayed about that command.

# DTM\_REVIEW> HELP

---

## Example

```
DTM_REVIEW> HELP DEFINE
```

```
DTM
```

```
REVIEW
```

```
DEFINE/KEY
```

Defines a keypad key to execute a command string.

Format:

```
DTM_REVIEW> DEFINE/KEY key-name "command-string" [/qualifier...
```

Additional information available:

Qualifiers

```
/ECHO /NOECHO /IF_STATE=(state-name,...) /NOIF_STATE
```

```
/LOCK_STATE /NOLOCK_STATE /SET_STATE=state-name
```

```
/NOSET_STATE /TERMINATE /NOTERMINATE
```

```
Parameters Command_Description Example
```

```
DTM REVIEW DEFINE/KEY Subtopic?
```

This example shows how to get help for the DEFINE/KEY command.

---

## DTM\_REVIEW> INSERT

Marks test descriptions for insertion into a group DEC/Test Manager creates when you exit from the Review subsystem.

---

### Format

**INSERT** [*result-description-expression*] [*/qualifier...*]

#### Command Qualifiers

/COMPARISON\_ABORTED  
/[NO]CONFIRM  
/NEW  
/NOT\_RUN  
/SUCCESSFUL  
/UNSUCCESSFUL  
/UPDATED

#### Defaults

Current result description  
/NOCONFIRM  
Current result description  
Current result description  
Current result description  
Current result description  
Current result description

---

### Command Parameter

#### *result-description-expression*

Specifies one or more test descriptions for insertion into a group that is created when you exit from the Review subsystem. The result description name for a test is the same as its test name. If you omit this parameter, DEC/Test Manager inserts into a group test descriptions from the current collection.

---

### Description

The INSERT command marks test descriptions for insertion into a group that DEC/Test Manager creates when you exit from the Review subsystem. All test descriptions selected with the INSERT command during a Review session are inserted into the same group. One group is created for each review session. To create a second group, exit from the Review subsystem, initiate another Review session for the same collection, and create another

## DTM\_REVIEW> INSERT

group. The EXIT/NOINSERT command specifies that the marked test descriptions are not to be inserted into a group when you exit from the Review subsystem. The group is not created.

DEC/Test Manager supplies the group name based on the name of the collection you are reviewing and the number of times it has been reviewed. The format is as follows:

collection-name\$DTM\_#

The number sign (#) specifies the number of times the collection has been reviewed.

Omitting the result description expression parameter marks the current test description for inclusion in the group. Specifying a result description name marks for inclusion that test description only. That result description becomes the current result description. Specifying a result description expression containing wildcard characters marks for inclusion all tests whose result description names match the result description expression. The current result description is not changed.

The comparison status qualifiers, /COMPARISON\_ABORTED, /NEW, /NOT\_RUN, /SUCCESSFUL, /UNSUCCESSFUL, and /UPDATED, mark for inclusion all result descriptions with the specified comparison status. Including one or more comparison status qualifiers and a result description expression marks for inclusion all test descriptions that match both the result description expression and one of the qualifiers.

---

## Command Qualifiers

### ***/COMPARISON\_ABORTED***

Marks for inclusion test descriptions for tests whose comparisons aborted. The default is to mark the test description associated with the current result description.

### ***/CONFIRM***

### ***/NOCONFIRM (D)***

Controls whether DEC/Test Manager prompts you to confirm the processing of each test description marked for insertion. Valid responses are Yes, No, All, or Quit.



## DTM\_REVIEW> INSERT

### ***/NEW***

Marks for inclusion test descriptions for new tests. The default is to mark the test description associated with the current result description.

### ***/NOT\_RUN***

Marks for inclusion test descriptions for tests that did not run. The default is to mark the test description associated with the current result description.

### ***/SUCCESSFUL***

Marks for inclusion test descriptions for tests that compared successfully. The default is to mark the test description associated with the current result description.

### ***/UNSUCCESSFUL***

Marks for inclusion test descriptions for tests that compared unsuccessfully. The default is to mark the test description associated with the current result description.

### ***/UPDATED***

Marks for inclusion test descriptions for tests whose benchmark files have been updated. The default is to mark the test description associated with the current result description.

---

## Example

```
DTM_REVIEW> INSERT MAIL_TEST
DTM-S-MRKFORINSERT, test_description MAIL_TEST marked for insertion
DTM_REVIEW> EXIT
DTM-S-CREATED, group RUNMAIL$DTM_1 created
DTM-S-EXIT, leaving Review subsystem
DTM>
```

This example inserts the `MAIL_TEST` test into a group that is created upon leaving the Review subsystem.

## DTM\_REVIEW> LAST

---

## DTM\_REVIEW> LAST

Moves you to the last result description in the collection you are reviewing.

---

### Format

#### LAST

Command Qualifiers	Defaults
None	None

---

### Description

The LAST command moves you to the last result description of the collection you are reviewing—the current collection. The last result description becomes the current result description.

---

### Example

```
DTM_REVIEW> LAST
```

```
.  
. .  
. .
```

This example displays the last result description in a collection.

---

## DTM\_REVIEW> NEXT

Moves you forward through the sequence of result descriptions being reviewed. Pressing the RETURN key also moves you forward to the next result description.

---

### Format

**NEXT** [*count*] [/*qualifier...*]

#### Command Qualifiers

/COMPARISON\_ABORTED  
/NEW  
/NOT\_RUN  
/SUCCESSFUL  
/UNSUCCESSFUL  
/UPDATED

#### Defaults

Next result description  
Next result description  
Next result description  
Next result description  
Next result description  
Next result description

---

### Command Parameter

#### *count*

An integer that indicates the number of result descriptions to move forward from the current result description. The default is 1.

---

### Description

The NEXT command moves you forward through the sequence of result descriptions being reviewed. The optional count parameter is the number of result descriptions to move forward from the current result description. If the parameter is omitted, you move forward to the next result description. You cannot use the NEXT command to move beyond the last result description. The result description to which you move forward becomes the current result description.

## DTM\_REVIEW> NEXT

The comparison status qualifiers */COMPARISON\_ABORTED*, */NEW*, */NOT\_RUN*, */SUCCESSFUL*, */UNSUCCESSFUL*, and */UPDATED* move you forward to the next result description with the specified comparison status. Specifying a count parameter in combination with a comparison status qualifier moves you forward through the specified number of result descriptions with the specified comparison status.

Pressing the RETURN key at the DTM\_REVIEW> prompt also moves you to the next result description.

---

## Command Qualifiers

### ***/COMPARISON\_ABORTED***

Moves you to the next result description for a test whose comparison aborted.

### ***/NEW***

Moves you to the next result description for a new test.

### ***/NOT\_RUN***

Moves you to the next result description for a test that did not run.

### ***/SUCCESSFUL***

Moves you forward to the next successfully compared result description.

### ***/UNSUCCESSFUL***

Moves you forward to the next unsuccessfully compared result description.

### ***/UPDATED***

Moves you forward to the next result description whose benchmark file has been updated.

---

## Example

```
DTM_REVIEW> NEXT 2  
.  
.  
.
```

This example displays the result description that is two positions ahead of the current result description.

## DTM\_REVIEW> PCA

---

## DTM\_REVIEW> PCA

Invokes the Analyzer of the VAX Performance and Coverage Analyzer (PCA)

---

### Format

#### PCA

Command Qualifiers	Defaults
None	None

---

### Description

The PCA command invokes the Analyzer of the VAX Performance and Coverage Analyzer (PCA).

The PCA command spawns a subprocess that invokes the Analyzer, specifies the default data file set up by DEC/Test Manager as input to the Analyzer, and sets up an Analyzer filter that includes only performance and coverage data collected while the current test was run. The subprocess spawned by the PCA command has all the globally defined symbols of the current process.

If the collection of tests you are reviewing was not run with the Collector, the Analyzer issues an error message when you enter the PCA command. The error message states that the expected data file does not exist. After the error message, you are returned to the Analyzer prompt (PCAA>). See the *Guide to VAX Performance and Coverage Analyzer* for information about the VAX Performance and Coverage Analyzer.

---

## DTM\_REVIEW> PRINT

Selects one or more files for printing.

---

### Format

**PRINT** [*result-description-expression*] [*/qualifier...*]

#### Command Qualifiers

/BENCHMARK  
/COMPARISON\_ABORTED  
/DIFFERENCES  
/[NO]LOG  
/NEW  
/NOT\_RUN  
/NOW  
/RESULT  
/SELECTED  
/SUCCESSFUL  
/UNSUCCESSFUL  
/UPDATED

#### Defaults

/RESULT  
Current result description  
/RESULT  
/LOG  
Current result description  
Current result description  
See text  
/RESULT  
See text  
Current result description  
Current result description  
Current result description

---

### Command Parameter

#### *result-description-expression*

Specifies one or more result descriptions from which the specified files are to be selected for printing. The result description name for a test is the same as its test name.

# DTM\_REVIEW> PRINT

---

## Description

The PRINT command selects one or more files for printing.

DEC/Test Manager informs you if the specified files do not exist. When you exit from the Review subsystem, the files you select are placed in the print queue as a single print job unless you specify otherwise. The /NOW qualifier selects all files specified on the current print command, concatenates them, and places them immediately in the print queue. The /SELECTED qualifier selects all files specified on all previous PRINT commands during this review session as well as all files specified on the current PRINT command, concatenates them, and places them immediately in the print queue.

Result and difference files that you selected for printing but subsequently deleted with an UPDATE command are not printed. The /NOPRINT qualifier specifies that the selected files are not to be submitted to the print queue.

Omitting the result description expression parameter causes DEC/Test Manager to select files from the current result description. Specifying a result description name causes DEC/Test Manager to select files from that result description only. That result description becomes the current result description.

Specifying a result description expression containing wildcard characters causes DEC/Test Manager to select files from all result descriptions whose result description names match the result description expression. The current result description is not changed.

The comparison status qualifiers, /COMPARISON\_ABORTED, /NEW, /NOT\_RUN, /SUCCESSFUL, /UNSUCCESSFUL, and /UPDATED, specify that DEC/Test Manager select files from result descriptions with the specified comparison statuses. Including one or more comparison status qualifiers and a result description expression causes DEC/Test Manager to select files from all result descriptions that match both the result description expression and one of the qualifiers. The comparison status qualifiers cannot be included with a result description name.

The output file qualifiers /BENCHMARK, /DIFFERENCES, and /RESULT specify that DEC/Test Manager select the specified files for printing. The default qualifier is /RESULT.



When DEC/Test Manager prints a benchmark file that is stored in a CMS library, the benchmark file is fetched and the fetched copy is deleted after it is printed.

---

## **Command Qualifiers**

### ***/BENCHMARK***

Prints benchmark files from the specified result descriptions. The default is */RESULT*.

### ***/COMPARISON\_ABORTED***

Prints files from result descriptions for tests whose comparisons aborted. The default is to select files from the test description associated with the current result description.

### ***/DIFFERENCES***

Prints difference files from the specified result descriptions. The default is */RESULT*.

### ***/LOG (D)***

#### ***/NOLOG***

Controls whether DEC/Test Manager displays informational and success messages on the screen.

### ***/NEW***

Prints files from result descriptions for new tests. The default is to select files from the test description associated with the current result description.

### ***/NOT\_RUN***

Prints files from result descriptions for tests that did not run. When a test does not run, its result description may or may not contain a benchmark file. The default is to select files from the test description associated with the current result description.

### ***/NOW***

Concatenates all files selected for printing on the current **PRINT** command and immediately places them in the print queue. The default is to place the concatenated files in the print queue when you exit from the Review subsystem.

## DTM\_REVIEW> PRINT

### ***/RESULT***

Prints the result file from the specified result descriptions. This is the default.

### ***/SELECTED***

Concatenates all files already selected for printing and the currently specified file and immediately places them in the print queue.

The default is to place the concatenated files in the print queue when you exit from the Review subsystem.

### ***/SUCCESSFUL***

Prints files from result descriptions for tests that compared successfully. The default is to select files from the test description associated with the current result description.

### ***/UNSUCCESSFUL***

Prints files from result descriptions for tests that compared unsuccessfully. The default is to select files from the test description associated with the current result description.

### ***/UPDATED***

Prints files from result descriptions whose benchmark files have been updated. The default is to select files from the test description associated with the current result description.

---

## Example

```
DTM_REVIEW> PRINT/DIFFERENCES
%DTM-S-PRINT, file DUA1:[USER01.DTMLIB.MAIL_COLLECTION]MAIL_TEST.DIF of
test MAIL_TEST selected for printing
DTM_REVIEW> EXIT
%DTM-S-PRINTQD, print job has been sent to the print queue
-DTM-I-TEXT, Job MAIL_TEST (queue SYSSPRINT, entry 32) started on SYSSPRINT
%DTM-S-EXIT, leaving Review subsystem
DTM>
```

This example shows how to queue the differences file for printing.

---

## DTM\_REVIEW> SELECT

Moves you to the specified result description.

---

### Format

**SELECT** *result-description-name*

**Command Qualifiers**

None

**Defaults**

None

---

### Command Parameter

***result-description-name***

Specifies the name of the selected result description. The result description name for a test is the same as its test name.

The result description name is required; you cannot specify a result description expression.

---

### Description

The **SELECT** command moves you to the specified result description. The specified result description becomes the current result description.

---

### Example

```
DTM_REVIEW> SELECT MAIL_TEST  
Result Description MAIL_TEST Comparison Status : Unsuccessful
```

This example displays the **MAIL\_TEST** result description in a collection.

## DTM\_REVIEW> SHOW

---

# DTM\_REVIEW> SHOW

Displays and describes output files for specified result descriptions.

---

## Format

**SHOW** [*result-description-expression*] [/*qualifier...*]

### Command Qualifiers

/BENCHMARK  
/COMPARISON\_ABORTED  
/DIFFERENCES  
/FILES  
/NEW  
/NOT\_RUN  
/OUTPUT[=file-specification]  
/RESULT  
/SUCCESSFUL  
/SUMMARY  
/UNSUCCESSFUL  
/UPDATED

### Defaults

/FILES  
Current result description  
/FILES  
/FILES  
Current result description  
Current result description  
/OUTPUT=SYS\$OUTPUT  
/FILES  
Current result description  
/FILES  
Current result description  
Current result description

---

## Command Parameter

### *result-description-expression*

Specifies one or more result descriptions about which the specified information is to be displayed. The result description name for a test is the same as its test name.

---

## **Description**

The **SHOW** command describes and displays output files for specified result descriptions. The **/BENCHMARK**, **/DIFFERENCES**, and **/RESULT** qualifiers display files for the specified result descriptions or display a message indicating that the file does not exist. Using these qualifiers with comparison status qualifiers displays the specified output files for result descriptions with the specified comparison status.

Omitting the result description expression parameter displays information about the current result description. Specifying a result description name displays information about that result description only. That result description becomes the current result description.

Specifying a result description expression containing wildcard characters displays information about each result description that matches the result description expression. The current result description does not change.

You can specify the comparison status qualifiers, **/COMPARISON\_ABORTED**, **/NEW**, **/NOT\_RUN**, **/SUCCESSFUL**, **/UNSUCCESSFUL**, and **/UPDATED**, only with a result description expression; you cannot use them with a result description name. Using comparison status qualifiers displays information about all result descriptions matching the result description expression and a comparison status qualifier.

The **/FILES** qualifier displays the comparison status for the specified result descriptions and, for each output file, states whether it exists. If a benchmark file exists, its file specification is also displayed. You cannot specify the **/FILES** qualifier with the output file qualifiers or with the **/SUMMARY** qualifier.

The **/SUMMARY** qualifier displays the Collection Summary Information. The **/SUMMARY** qualifier cannot be specified with the **/FILES** qualifier, with the output file qualifiers, or with the comparison status qualifiers.

DEC/Test Manager provides you with a set of default keypad definitions for positioning benchmark, result, or difference screen images when you use the Review subsystem **SHOW/BENCHMARK**, **SHOW/RESULT**, or **SHOW/DIFFERENCES** commands; see Chapter 5 for more information.

## DTM\_REVIEW> SHOW

You can use the **DEFINE/KEY** command to replace these definitions or to define the undefined keypad keys. Pressing the PF2 key displays the default key definitions. See the DEC/Test Manager Review Subsystem **DEFINE/KEY** command section for more information about defining keys.

---

### Command Qualifiers

#### ***/BENCHMARK***

Displays the benchmark file. The default is to state whether output files (benchmark, difference, and result files) exist. If the benchmark file exists, its file specification is displayed.

For a noninteractive test, DEC/Test Manager displays the benchmark file. For an interactive test, DEC/Test Manager displays the benchmark file screen by screen and provides you with a keypad for manipulating the file. The keypad and the procedure for using it are described in Chapter 5.

If the benchmark file is in a CMS library, DEC/Test Manager fetches the benchmark.

If you include a comparison status qualifier with the **/BENCHMARK** qualifier, DEC/Test Manager displays the benchmark file for result descriptions with the specified comparison status.

#### ***/COMPARISON\_ABORTED***

States whether output files (benchmark, difference, and result files) exist for the specified result descriptions with the comparison aborted comparison status. The default is to display information for the current result description. If the benchmark file exists, its file specification is displayed.

If you include an output file qualifier with the **/COMPARISON\_ABORTED** qualifier, the specified output files are displayed for result descriptions with the comparison aborted comparison status. You cannot specify a comparison status qualifier with a result description name parameter; a result description expression is required.

#### ***/DIFFERENCES***

Displays the difference file. The default is to state whether output files (benchmark, difference, and result files) exist. If the benchmark file exists, its file specification is displayed.

For a noninteractive test, DEC/Test Manager displays the difference file. For an interactive test, DEC/Test Manager displays the benchmark and result files screen by screen with differences marked and provides you with a keypad for manipulating the files. The keypad and the procedure for using it are described in Chapter 5.

If you include a comparison status qualifier with the `/DIFFERENCES` qualifier, DEC/Test Manager displays the difference file for result descriptions with the specified comparison status.

## **`/FILES`**

Displays the comparison status for result descriptions and states whether output files (benchmark, difference, and result files) exist for the result description. The default is `/FILES`. If the benchmark file exists, its file specification is displayed.

You cannot include the `/FILES` qualifier with the included `/BENCHMARK`, `/DIFFERENCES`, `/RESULT`, and `/SUMMARY` qualifiers.

## **`/NEW`**

States whether output files (benchmark, difference, and result files) exist for result descriptions with the new comparison status. The default is to display information for the current result description.

If you include an output file qualifier with the `/NEW` qualifier, the specified output files are displayed for result descriptions with the new comparison status. If a test is new, it has a result file but it does not have a benchmark or difference file. You cannot specify a comparison status qualifier with a result description name parameter; a result description expression is required.

## **`/NOT_RUN`**

States whether output files (benchmark, difference, and result files) exist for result descriptions with the not run comparison status. The default is to display information for the current result description. If the benchmark file exists, its file specification is displayed.

If you include an output file qualifier with the `/NOT_RUN` qualifier, the specified output files are displayed for result descriptions with the not run comparison status. If a test does not run, it does not have a result or difference file. It might have a benchmark file. You cannot specify a comparison status qualifier with a result description name parameter. A result description expression is required.

## DTM\_REVIEW> SHOW

### ***/OUTPUT[=file-specification]***

Sends the requested output to the specified file. The default is SYS\$OUTPUT.

### ***/RESULT***

Displays the result file. The default is to state whether output files (benchmark, difference, and result files) exist.

For a noninteractive test, DEC/Test Manager displays the result file. If the benchmark file exists, its file specification is displayed.

For an interactive test, DEC/Test Manager displays the result file screen by screen and provides you with a keypad for manipulating the file. The keypad and the procedure for using it are described in Chapter 5.

If you include a comparison status qualifier with the */RESULT* qualifier, DEC/Test Manager displays the result file for result descriptions with the specified comparison status.

### ***/SUCCESSFUL***

States whether output files (benchmark, difference, and result files) exist for result descriptions with the successful comparison status. The default is to display information for the current result description. If the benchmark file exists, its file specification is displayed.

If you include an output file qualifier with the */SUCCESSFUL* qualifier, the specified output files are displayed for result descriptions with the successful comparison status. If a test is successful, it has a benchmark file. Its result file was deleted and no difference file was created. You cannot specify a comparison status qualifier with a result description name parameter; a result description expression is required.

### ***/SUMMARY***

Displays the Collection Summary Information (the information displayed when you first enter the Review subsystem). The default is */FILES*. You cannot specify the */SUMMARY* and */FILES* qualifiers with the same command.

The */SUMMARY* qualifier is mutually exclusive with all qualifiers except the */OUTPUT* qualifier.



## ***/UNSUCCESSFUL***

States whether output files (benchmark, difference, and result files) exist for result descriptions with the unsuccessful comparison status. The default is to display information for the current result description. If the benchmark file exists, its file specification is displayed.

If you include an output file qualifier with the */UNSUCCESSFUL* qualifier, the specified output files are displayed for result descriptions with the unsuccessful comparison status. (If a test is unsuccessful, it has benchmark, result, and difference files.) You cannot specify a comparison status qualifier with a result description name parameter; a result description expression is required.

## ***/UPDATED***

States whether output files (benchmark, difference, and result files) exist for result descriptions with the updated comparison status. The default is to display information for the current result description. If the benchmark file exists, its file specification is displayed.

If you include an output file qualifier with the */UPDATED* qualifier, the specified output files are displayed for result descriptions with the updated comparison status. An updated test is a test whose benchmark file was created from its result file since the time when the test was last executed. An updated test does not have a result or difference file. You cannot specify a comparison status qualifier with a result description name parameter; a result description expression is required.

---

## **Example**

```
DTM_REVIEW> SHOW/UNSUCCESSFUL
Result Description YYY      Comparison Status : Unsuccessful

      Benchmark File is DUA1:[USER01.DTM.DTMLIB]YYY.BMK
      Result file is present
      Difference file is present

DTM_REVIEW>
```

This example displays all the unsuccessful result descriptions in a collection.

## DTM\_REVIEW> SPAWN

---

## DTM\_REVIEW> SPAWN

Creates a subprocess of the current DEC/Test Manager session.

---

### Format

**SPAWN** *[command] [/qualifier...]*

#### Command Qualifiers

**/CARRIAGE\_CONTROL**  
**/[NO]CLI[=cli]**  
**/INPUT=file-specification**  
**/[NO]KEYPAD**  
**/[NO]LOGICAL\_NAMES**  
**/[NO]NOTIFY**  
**/OUTPUT=file-specification**  
**/PROCESS=subprocess-name**  
**/[NO]PROMPT[=string]**  
**/[NO]SYMBOLS**  
**/[NO]WAIT**

#### Defaults

See text  
See text  
See text  
**/KEYPAD**  
**/LOGICAL\_NAMES**  
**/NONOTIFY**  
See text  
See text  
See text  
**/SYMBOLS**  
**/WAIT**

---

### Command Parameter

#### *command*

Specifies an optional command to be executed by the subprocess you are creating. If you specify the command parameter, you create a subprocess that executes the command and returns control to the DEC/Test Manager session when the command terminates. If you include the **/INPUT** qualifier with the command parameter, the subprocess reads commands from the specified input file after the command string executes. The command string can be up to 132 characters.

If you omit the command parameter, the **SPAWN** command creates a subprocess and attaches the terminal to it. You can return to the DEC/Test Manager session by logging out of the subprocess or by issuing the

**ATTACH/PARENT** command. If you have created several subprocesses, you can switch between them using the **ATTACH/IDENTIFICATION** command.

---

### Description

The SPAWN command creates a subprocess of the current DEC/Test Manager session (the parent process). The context of the DEC/Test Manager session is copied to the subprocess.

You can use the SPAWN command to leave the Review subsystem temporarily, to create another DEC/Test Manager session, or to edit a file, and then return to the original Review session.

---

### Command Qualifiers

***/CARRIAGE\_CONTROL***

***/NOCARRIAGE\_CONTROL***

Determines whether carriage control or line feed characters or both are prefixed to the prompt string of the subprocess. The default is the current setting of the parent process.

***/CLI[=cli]***

***/NOCLI***

Specifies an alternate command language interpreter (CLI) for the subprocess to use. The CLI you specify must be located in SYS\$SYSTEM and have the file type .EXE. The default is the CLI the parent process uses.

***/INPUT=file-specification***

Specifies an input file containing one or more commands for the spawned subprocess to execute. If you specify a command with an input file, the command is processed before the commands in the input file. The subprocess terminates when processing is complete. You cannot use wildcards in the file specification.

## DTM\_REVIEW> SPAWN

***/KEYPAD (D)***

***/NOKEYPAD***

Determines whether DCL keypad symbols and the current DCL keypad state are copied from the DCL keypad in the parent process to the subprocess. Use the */NOKEYPAD* qualifier if you do not want the key settings to be copied.

***/LOGICAL\_NAMES (D)***

***/NOLOGICAL\_NAMES***

Determines whether the system passes process logical names and logical name tables to the subprocess, except those marked *CONFINE* or created in executive or kernel mode.

***/NOTIFY***

***/NONOTIFY (D)***

Determines whether a message is sent to the terminal to notify you that the subprocess has completed or aborted. Do not specify */NOTIFY* unless you also specify the */NOWAIT* qualifier.

***/OUTPUT=file-specification***

Specifies the output file to which the output of the SPAWN operation is to be written. The default is to direct the output to the current *SYS\$OUTPUT* device. When you specify */NOWAIT*, you should use */OUTPUT* to specify an output other than *SYS\$OUTPUT* to prevent the terminal from being used by both processes simultaneously.

***/PROCESS=subprocess-name***

Specifies the name of the subprocess to be created. The default is *USERNAME\_n* (where *n* denotes a unique number).

***/PROMPT[=string]***

Specifies the DCL-prompt string for the subprocess. The default is to copy the current prompt string from the parent process. If you specify */PROMPT* but do not specify a string, the default prompt is displayed.

***/SYMBOLS (D)***

***/NOSYMBOLS***

Determines whether the system passes DCL global and local symbols to the subprocess.

***/WAIT (D)***

***/NOWAIT***

Controls whether the system waits until the subprocess is completed before enabling more commands to be issued to the parent process. The ***/NOWAIT*** qualifier enables you to enter more commands while the specified subprocess is running. When you specify ***/NOWAIT***, you should also specify ***/OUTPUT*** to direct output to a file (rather than to the screen). This prevents the terminal from being used by both processes simultaneously.

---

### Example

```
DTM_REVIEW> SPAWN MAIL
```

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

```
MAIL>
```

This example spawns the VMS MAIL Utility from the DEC/Test Manager subsystem. Enter the **ATTACH** command to terminate the MAIL session and return to the DEC/Test Manager system level.

## DTM\_REVIEW> UPDATE

---

## DTM\_REVIEW> UPDATE

Makes the result file for the specified result descriptions the new benchmark file and deletes the previous benchmark file if it resides in the DEC/Test Manager library.

---

### Format

**UPDATE** [*result-description-expression*] [/qualifier...] "remark"

#### Command Qualifiers

/[NO]CONFIRM  
/[NO]LOG

#### Defaults

See text  
/LOG

---

### Command Parameters

#### *result-description-expression*

Specifies one or more result descriptions whose benchmark files are to be updated. The result description name for a test is the same as its test name.

#### *remark*

Specifies a string that contains a comment. You must specify a remark within quotation marks (" "). The exception to this rule is when you specify a remark string at the remark prompt. If you do not provide a remark string, you are prompted for one. However, a null remark is permitted.

---

### Description

The UPDATE command creates a new benchmark file from the existing result file for the specified result descriptions. If the previous benchmark file is in the DEC/Test Manager library, it is deleted. On DECwindows benchmark image files, masked areas are transferred to the new benchmark file when it is updated.

To be updated, the result description must have a comparison status of comparison aborted, unsuccessful, or new. You cannot update the benchmark file for a result description with a comparison status of successful, not run, or updated. DEC/Test Manager automatically deletes the result file for successful tests.

Omitting the result description expression parameter causes DEC/Test Manager to update the benchmark file for the current result description, if its comparison status is comparison aborted, new, or unsuccessful. Specifying a result description name causes DEC/Test Manager to update the benchmark file for that result description only, providing that its comparison status is comparison aborted, new, or unsuccessful. This result description becomes the current result description.

Specifying a result description expression containing wildcard characters causes DEC/Test Manager to update benchmark files for all test descriptions whose result description names match the result description expression and whose comparison status is comparison aborted, new, or unsuccessful. The current result description is not changed.

If you store the benchmark files in a CMS library, the UPDATE command issues the CMS RESERVE and REPLACE commands to replace the old benchmark file. The result file is used as input to the CMS REPLACE command. If you use CMS classes for the benchmark files, the UPDATE command also issues the CMS INSERT command with the /SUPERSEDE qualifier to place the current generation of the benchmark file into the class. If DEC/Test Manager is updating a generation other than the latest, it creates a variant line designated "D".

When you update a benchmark file in a CMS library for a new test with no existing benchmark file, DEC/Test Manager creates a new element in the specified CMS library. If you also specify a class, the new element is also inserted into the class.

Use the UPDATE command only when you are sure you want to delete the current benchmark file and replace it with the current result file. This procedure is irreversible; therefore, carefully consider the possible effects before using the UPDATE command.

You can also replace benchmark files with the MODIFY TEST\_DESCRIPTION command with the /BENCHMARK=file-name or /NOBENCHMARK qualifier.

# DTM\_REVIEW> UPDATE

---

## Command Qualifiers

***/CONFIRM***

***/NOCONFIRM (D)***

Controls whether DEC/Test Manager displays each test name before updating it and prompts you to confirm whether you want the test results updated. If you specify a wildcard result description expression, DEC/Test Manager automatically prompts you.

***/LOG (D)***

***/NOLOG***

Controls whether DEC/Test Manager displays informational and success messages on your screen.

---

## Example

```
DTM_REVIEW> SELECT MSGTEST
Result Description MSGTEST      Comparison Status : Unsuccessful

DTM_REVIEW> UPDATE
%DTM_I_UPDATED, the benchmark for test MSGTEST has been updated
DTM_REVIEW>
```

This example selects the MSGTEST test results and then uses the UPDATE command to replace the current benchmark file with the test results.



# DEC/Test Manager Messages

---

This appendix lists the DEC/Test Manager messages alphabetically. The messages are accompanied by explanations and, where applicable, suggested actions to recover from errors.

---

## A.1 Message Display

DEC/Test Manager messages are displayed on the current output device. For an interactive user, this device is a terminal or workstation. If DEC/Test Manager is run in batch mode, messages are written to the log file.

---

### A.1.1 Message Format

A DEC/Test Manager message has the following format:

%DTM-severity\_code-message\_name, text of message

Table A-1 shows how the message fields are interpreted.

**Table A-1: DEC/Test Manager Message Fields**

Component	Description
%DTM	A prefix indicating that the message originates from DEC/Test Manager
Severity_code	A single letter indicating one of the five codes described in Section A.1.2
Message_name	A name that uniquely identifies the message
Text of message	A one- or two-sentence description of an event that has occurred

---

## A.1.2 Severity Codes

DEC/Test Manager issues messages with varying severity levels. The severity level of a message indicates the general nature of the message and is represented by one of the codes in Table A-2.

**Table A-2: DEC/Test Manager Message Severity Codes**

<b>Code</b>	<b>Description</b>
S (Successful)	Indicates that DEC/Test Manager has performed the request.
I (Information)	Indicates certain kinds of information about the command you issued. For example, DEC/Test Manager informs you if it is waiting for the database to become available.
W (Warning)	Indicates that DEC/Test Manager has encountered a minor conflict, but one that does not stop processing of your command.
E (Error)	Indicates that DEC/Test Manager is unable to perform the requested command; you must correct the problem and enter the command again. Processing of the command might continue.
F (Fatal)	Indicates that DEC/Test Manager is about to terminate because of a problem that prevents it from continuing any further. Processing of the command stops.

Some fatal error problems can be resolved by issuing a `VERIFY` command with the `/RECOVER` qualifier; see Chapter 7 for more information.

---

## A.2 DEC/Test Manager Messages

This section lists the DEC/Test Manager messages with a brief explanation of each message and the recommended user action. If no user action is required, this section is omitted. The messages are listed in alphabetical order, by message name. A term enclosed in single quotation marks is variable information and DEC/Test Manager substitutes appropriate information when the message is displayed on the screen or in the log file.

### NOTE

Every message begins with `%DTM` and a one-letter severity code before the message name.

ABORTING, job 'job-name' (entry 'number') is aborting in queue  
'queue-name'

**Explanation:** The execution of the specified job was stopped, and the job is in the process of aborting.

**User Action:** Wait for the job to finish aborting before trying to use the collection.

ABSTIM, 'qualifier' time value must be absolute

**Explanation:** Specify an absolute time value for the /BEFORE and /SINCE qualifiers.

**User Action:** Correct the time value and enter the command again.

ACCVIO, access violation reading routine argument at virtual address  
'address'

**Explanation:** DEC/Test Manager could not access a routine argument passed to it by the callable interface.

**User Action:** Correct the call to DTM\$DTM.

ALLOW, could not allocate structure

**Explanation:** DEC/Test Manager could not allocate an internal structure.

**User Action:** Check quotas and SYSGEN parameters. See the secondary messages for more information.

ALPHACHAR, the first character in 'expression' must be alphanumeric

**Explanation:** You specified an expression that does not begin with a letter or a number.

**User Action:** Correct the expression and enter the command again.

ALRDYCOMPARED, 'type' (collection or test) 'name' has already been  
compared

**Explanation:** If the 'type' is a collection, then you cannot recompare a collection once it has been reviewed. If the 'type' is a test, then the test was compared previously and will not be compared again.

ALRDYEXISTS, 'name' is already an 'object-name'

**Explanation:** You specified a name for a collection, group, test description, or variable that already exists in the current library.

**User Action:** Select another name and enter the command again.

ALRDYINGRP, object 'name' is already in group 'name'

**Explanation:** The test description or group already belongs to the specified group.

ALRDYUPD, the benchmark for result description 'name' has already been updated

**Explanation:** You tried to update a benchmark file that has already been updated.

ANSI\_CRT, recording terminal was an ANSI crt, display terminal is not

**Explanation:** The terminal on which you are monitoring an executing test has characteristics different from the terminal on which the test was recorded.

**User Action:** DEC/Test Manager is executing the test correctly (even though the display may look incorrect) and will create a proper result file. Use the Review subsystem SHOW/RESULT command to display the result file.

ASSIGNERR, unable to assign an I/O channel to device

**Explanation:** DEC/Test Manager could not access the specified device.

**User Action:** Check your quotas and SYSGEN parameters, make sure the device is available, and check the device's protection and ownership.

ASTERR, error declaring an AST

**Explanation:** A system service failed to create an Asynchronous System Trap (AST).

**User Action:** Check the AST quotas.

ASTERROR, AST routine received the following error:

**Explanation:** An error occurred at Asynchronous System Trap (AST) level.

**User Action:** See the secondary messages for more information.

ASTRONLY, the only parameter recognized by this command is '\*'

**Explanation:** You entered a parameter other than an asterisk (\*). If you enter any parameter at all, it must be the asterisk wildcard character.

**User Action:** Enter the command again with no parameter or with an asterisk (\*).

ATTACHERR, could not put pseudoterminal in ATTACH mode

**Explanation:** DEC/Test Manager could not put the PTY device into ATTACH mode.

**User Action:** Submit a Software Performance Report (SPR).

AVO\_TERM, recording terminal had AVO option, display terminal does not

**Explanation:** The terminal on which you are monitoring an executing test has characteristics different from the terminal on which the test was recorded.

**User Action:** DEC/Test Manager is executing the test correctly (even though the display may look incorrect) and will create a proper result file. Use the Review subsystem SHOW/RESULT command to display the result file.

BADCOLL, there is something wrong with collection 'collection-name'

**Explanation:** There is a problem with the specified collection.

**User Action:** Delete the collection or, if the collection is required, restore the library from a backup tape.

BADCTRLRECORD, Playback encountered an invalid CONTROL record in memory

**Explanation:** This is an internal DEC/Test Manager DECwindows condition.

**User Action:** Submit a Software Performance Report (SPR).

BADLENSTR, 'type' block length is 'length', should be 'length'

**Explanation:** DEC/Test Manager has discovered an inconsistency or error in your DEC/Test Manager library that it cannot correct.

**User Action:** Restore the library from a backup tape to ensure that you are using a consistent library.

**BADLIB**, there is something wrong with your DEC/Test Manager library

**Explanation:** DEC/Test Manager has discovered an inconsistency or error in your DEC/Test Manager library that it cannot correct.

**User Action:** Restore the library from a backup tape to ensure that you are using a consistent library.

**BADORDSTR**, 'type' block 'name' is out of order

**Explanation:** DEC/Test Manager has discovered an inconsistency or error in your DEC/Test Manager library that it cannot correct.

**User Action:** Restore the library from a backup tape to ensure that you are using a consistent library.

**BADPROCESS**, error processing file\_spec at line ##

**Explanation:** An error has been found processing a DECwindows session file.

**User Action:** Investigate the error, and submit a Software Performance Report (SPR).

**BADPROTOCOL**, corrupted protocol packet

**Explanation:** DEC/Test Manager encountered an unrecognized DECwindows protocol packet.

**BADPTR**, 'type' block has address 'address' outside range of database

**Explanation:** DEC/Test Manager has discovered an inconsistency or error in your DEC/Test Manager library that it cannot correct.

**User Action:** Restore the library from a backup tape to ensure that you are using a consistent library.

**BADSESSION**, session file has been corrupted

**Explanation:** The session file is corrupt.

**User Action:** Rerecord the session file.

**BADSESSIONREAD**, Error reading session file file\_spec at line ##

**Explanation:** The DECwindows playback system failed to read the session file at the line specified.

**User Action:** Check for a corrupted session file. Extract and restore the session file to verify its contents. Submit a Software Performance Report (SPR).

BADSTATUS, UNKNOWN error status returned from load\_extension()

**Explanation:** This is an internal DEC/Test Manager DECwindows condition.

**User Action:** Submit a Software Performance Report (SPR).

BADTYPSTR, 'type' block type is 'identifier', it should be 'identifier'

**Explanation:** DEC/Test Manager has discovered an inconsistency or error in your DEC/Test Manager library that it cannot correct.

**User Action:** Restore the library from a backup tape to ensure that you are using a consistent library.

BADVERSTR, 'type' block version is 'identifier', it should be 'identifier'

**Explanation:** DEC/Test Manager has discovered an inconsistency or error in your DEC/Test Manager library that it cannot correct.

**User Action:** Restore the library from a backup tape to ensure that you are using a consistent library.

BADWRITE, Fatal error writing output file file\_spec while processing input file line ##

**Explanation:** A fatal error has occurred while writing the DECwindows binary session file.

**User Action:** Investigate the error, and submit a Software Performance Report (SPR).

BCKPTRSTR, 'type' back pointer is 'identifier' previous block is 'identifier'

**Explanation:** DEC/Test Manager has discovered an inconsistency or error in your DEC/Test Manager library that it cannot correct.

**User Action:** Restore the library from a backup tape to ensure that you are using a consistent library.

BEGIN, your interactive test session is now beginning...

**Explanation:** Your recording session is beginning.

BMK\_NOTSAVED, no benchmark file will be saved

**Explanation:** Your benchmark file will not be saved.

BMK\_SAVED, benchmark has been saved in file 'name'

**Explanation:** Your benchmark file has been saved.

CANTCREATETASK, cannot create new task

**Explanation:** DEC/Test Manager was unable to create a task.

**User Action:** Check subprocess quota.

CANTSETMODMAP, cannot set modifier mappings

**Explanation:** DEC/Test Manager cannot set or restore keyboard modifier mappings when the Lock key is pressed.

**User Action:** Release the Lock key and try the operation again.

CANTOPEN, cannot open display on requested device 'device-name'

**Explanation:** DEC/Test Manager cannot open a connection to the DECwindows server.

**User Action:** Check the translation of the DECW\$DISPLAY logical name.

CANTRESUB, cannot resubmit this collection

**Explanation:** You cannot resubmit this collection.

CFREEFAIL, Unable to cfree a data\_structure

**Explanation:** This is an internal DEC/Test Manager DECwindows condition.

**User Action:** Submit a Software Performance Report (SPR).

CHECKCMS, check that SYS\$SHARE:CMSPROSHR.EXE is installed

**Explanation:** The VAX Code Management System (CMS) shareable image CMSSHR.EXE could not be activated.

**User Action:** Verify that the CMS startup file was executed when your system booted and that SYS\$SHARE:CMSPROSHR.EXE is installed with the /OPEN/SHARE/PROTECTED qualifiers. Consult with your system manager.

CLEANUPERR, Clean\_up Error clean\_up operation

**Explanation:** This is an internal DEC/Test Manager DECwindows condition.

**User Action:** Submit a Software Performance Report (SPR).



CLREFERR, unable to clear a system event flag

**Explanation:** A system service failed to clear an event flag.

**User Action:** Check your quotas and SYSGEN parameters. Consult with your system manager.

CMDFILABORT, aborting execution of commands from command file  
'name'

**Explanation:** An error occurred while DEC/Test Manager was executing commands from your start-up command file.

**User Action:** Correct the command that caused the error.

CMDNOTALLOWED, 'command' command is not allowed when  
Reviewing in Read\_only mode

**Explanation:** You attempted to enter an INSERT or UPDATE command while reviewing a collection in read-only mode.

**User Action:** Exit the Review subsystem and then reenter it without the /READ\_ONLY qualifier. As the primary reviewer, you can issue the INSERT and UPDATE commands.

CMDTOOLONG, command over 255 characters long, RECREATE 'name'  
will not work

**Explanation:** The original CREATE COLLECTION command was more than 255 characters long. Due to restrictions in the command line interface (CLI), you cannot currently use the RECREATE command for this collection.

**User Action:** Use the DELETE COLLECTION command to manually delete the existing collection. Then reenter the original CREATE COLLECTION command.

CMPSTATIGNORED, comparison status qualifier ignored

**Explanation:** You specified both a result-description-name parameter and one or more comparison status qualifiers on a command. DEC/Test Manager ignored the comparison status qualifiers.

CMSTRYAGNLAT, could not lock CMS library 'library-name'

**Explanation:** DEC/Test Manager attempted to lock a CMS library but could not because the library was in use.

**User Action:** Reissue the command when the library is no longer in use.

CNTRLABORT, Abort with Clean Up...

**Explanation:** This message is seen when the user issues a Control-Y or Control-C from the terminal where DEC/Test Manager is doing a DECwindows based operation.

CNTSTR, block 'type' count is 'actual-count', it should be 'correct-count'

**Explanation:** The specified block type count is incorrect.

**User Action:** Use the VERIFY/RECOVER command to restore your library to a usable state.

CNVNOTNEC, conversion not necessary, library already version 'number'

**Explanation:** The old library you specified on the CONVERT LIBRARY command does not need to be converted.

COLLINUSE, collection 'name' is in use

**Explanation:** The collection you specified is currently in use.

**User Action:** Wait and enter the command again when the collection is not in use. If you see this message for a collection that you know is not in use, there is a problem with the collection. Use the VERIFY/RECOVER command to restore your library to a usable state.

COLNOTCOMP, collection 'name' has not been compared

**Explanation:** You tried to review a collection that has not yet been compared.

**User Action:** Compare the collection and enter the command again.

COLNOTRUN, collection 'name' has not been run

**Explanation:** You tried to compare or review a collection that has not yet been executed.

**User Action:** Execute the collection and enter the command again after the collection has finished executing.

COLNOTRVW, collection 'name' has not been reviewed

**Explanation:** You tried to resubmit or re-create a collection that has not been reviewed.

**User Action:** Review the collection and enter the command again

COLSTATERR, collection is in an inconsistent state

**Explanation:** The collection is in an unusable state.

**User Action:** Use the VERIFY/RECOVER command to restore the collection to a usable state.

COLSTOPPED, collection 'name' has been stopped

**Explanation:** DEC/Test Manager stopped execution of the specified collection.

COMPARED, collection 'name' compared

**Explanation:** DEC/Test Manager compared the specified collection.

CONCLUDED, your interactive test session has concluded

**Explanation:** You terminated the recording session.

CONDTRACE, routine\_name called routine\_name which returned

**Explanation:** This is an internal DEC/Test Manager DECwindows condition.

**User Action:** Submit a Software Performance Report (SPR) specifying this and any additional error messages that are seen.

CONTROLC, operation aborted by CTRL/C

**Explanation:** You pressed CTRL/C to abort an operation in progress.

**User Action:** Enter the command again to restart the operation.

CONFIGIOFAIL, Config\_Io error returned from extension

**Explanation:** This is an internal DEC/Test Manager DECwindows condition.

**User Action:** Submit a Software Performance Report (SPR).

CONNECTFAIL, Connection FAILED to Node node\_name

**Explanation:** DEC/Test Manager could not establish a DECwindows connection to the specified workstation.

**User Action:** Define DECW\$DISPLAY to be the nodename of the workstation that is being used. This often solves subprocess failure conditions when DEC/Test Manager is given a command to spawn for DECwindows recording.

CONVERTED, version 'number' library successfully converted to version 'number'

**Explanation:** DEC/Test Manager converted the library.

COPIED, test description 'name' copied

**Explanation:** DEC/Test Manager copied the specified test description.

CREATED, 'object' 'name' created

**Explanation:** DEC/Test Manager created the specified collection, group, test description, variable, or library.

CR\_FILLS, recording terminal needs no crfill, display terminal does

**Explanation:** The terminal on which you are monitoring an executing test has characteristics different from the terminal on which the test was recorded.

**User Action:** DEC/Test Manager is executing the test correctly (even though the display may look incorrect) and will create a proper result file. Use the Review subsystem SHOW/RESULT command to display the result file.

CTRLCERR, error enabling CTRL/C ASTs

**Explanation:** DEC/Test Manager could not enable CTRL/Cs on your terminal.

**User Action:** Check your terminal's protection, existence, and ownership.

CURRCOMPARE, collection 'name' is currently being compared

**Explanation:** You tried to compare a collection that is currently being compared.

**User Action:** Wait and enter the command again after the collection has been compared.

CURRTERM, characteristics of current terminal will be used

**Explanation:** DEC/Test Manager could not extract the terminal characteristics information from the specified session file because it could not find the file. DEC/Test Manager will use the terminal characteristics of your current terminal.

DASSGNERR, unable to deassign the I/O channel to ' device'

**Explanation:** A system service could not release control of the specified device.

**User Action:** Check the availability, protection, and ownership of the device.

DCTRL, error disabling control characters

**Explanation:** A system service failed to disable CTRL/Y handling.

**User Action:** See the secondary messages for more information.

DEALLOC, failed to deallocate ' structure'

**Explanation:** An internal structure could not be deallocated.

**User Action:** See the secondary messages for more information.

DEC\_CRT1, recording terminal was a DEC crt, display terminal is not

**Explanation:** The terminal on which you are monitoring an executing test has characteristics different from the terminal on which the test was recorded.

**User Action:** DEC/Test Manager is executing the test correctly (even though the display may look incorrect) and will create a proper result file. Use the Review subsystem SHOW/RESULT command to display the result file.

DEC\_CRT2, recording terminal was a DEC crt, display terminal is not

**Explanation:** The terminal on which you are monitoring an executing test has characteristics different from the terminal on which the test was recorded.

**User Action:** DEC/Test Manager is executing the test correctly (even though the display may look incorrect) and will create a proper result file. Use the Review subsystem SHOW/RESULT command to display the result file.

DECW\_NETFAIL, Unable to communicate with the DECwindows server

**Explanation:** DEC/Test Manager is unable to communicate with the DECwindows server due to a network failure.

**User Action:** Contact your system manager.

DEFAULTDIR, default directory cannot be a DEC/Test Manager library or any of its subdirectories

**Explanation:** You cannot set your default directory to the DEC/Test Manager library or to any of its subdirectories.

**User Action:** Change your default directory to be different from the DEC/Test Manager library or any of its subdirectories, and enter the command again.

DEFAULTED, 'type' file name defaulted to 'name'

**Explanation:** DEC/Test Manager provided a default file name or type.

**User Action:** Check the default name or type to verify that it agrees with the existing file name or type.

DEFCANCEL, default 'item' canceled

**Explanation:** DEC/Test Manager canceled your current default benchmark directory, template directory, collection prologue file, or collection epilogue file.

DELETED, 'object' 'name' deleted

**Explanation:** DEC/Test Manager deleted the specified collection, group, test description, or variable.

DELETIONS, 'count' deletion(s) completed

**Explanation:** DEC/Test Manager deleted the indicated number of items.

DELSINRUN, test description 'name' has been deleted since this collection was run

**Explanation:** DEC/Test Manager cannot insert this test description into the group created from the Review subsystem because the test description has been deleted.

DEVUNKNOWN, device type unknown—VT100 assumed

**Explanation:** Your terminal is of an unknown device type. DEC/Test Manager will record the interactive terminal session as if you were using a VT100 series terminal.

DIFFERENT, files for test 'name' are different

**Explanation:** The COMPARE command detected differences between the result and benchmark files for the specified test description.

DIREXISTS, directory 'name' already exists

**Explanation:** DEC/Test Manager could not create the specified collection. The collection subdirectory it tried to create already exists.

DISPLAYSUP, display will be suppressed since device 'name' is not a terminal

**Explanation:** You entered a PLAY or RUN command with SYS\$OUTPUT specifying a device other than a terminal. The test will execute properly, but you will not be able to monitor its progress visually.

DUPLICATETASK, duplicate task name

**Explanation:** An attempt was made to create a duplicate task. DEC/Test Manager assigns task names based on the operation being performed and the object on which the operation is being performed.

**User Action:** Wait for the original task to terminate.

ECTRL, error enabling control characters

**Explanation:** A system service failed to enable CTRL/Y handling.

**User Action:** See the secondary messages for more information.

EIGHTBIT, recording terminal handles 8-bit, display terminal does not

**Explanation:** The terminal on which you are monitoring an executing test has characteristics different from the terminal on which the test was recorded.

**User Action:** DEC/Test Manager is executing the test correctly (even though the display may look incorrect) and will create a proper result file. Use the Review subsystem SHOW/RESULT command to display the result file.

EMPTYGROUP, group 'name' contains no test descriptions

**Explanation:** The specified group contains no test descriptions.

**User Action:** Enter the command again without the named group, or examine the library to see if this group should contain test descriptions.

ENDPTRSTR, 'type' end pointer is 'identifier', last block is 'identifier'

**Explanation:** The last block type is incorrect.

**User Action:** Use the VERIFY/RECOVER command to restore your library to a usable state.

ERRCOMP, error comparing test 'name'

**Explanation:** DEC/Test Manager could not compare the result and benchmark files for the specified test.

**User Action:** See the secondary message for more information.

ERRCREASSOC, error associating variable 'name' with test description 'name'

**Explanation:** DEC/Test Manager could not associate the specified variable and test description.

**User Action:** See the secondary message for more information.

ERRCRETASK, task not created

**Explanation:** An error occurred while DEC/Test Manager was creating a task so that it could not create the task.

**User Action:** See secondary message for more information.

ERRDELETIONS, 'count' deletion(s) completed with 'count' errors

**Explanation:** DEC/Test Manager encountered one or more errors while deleting items.

ERRDELFIL, error deleting file 'name'

**Explanation:** DEC/Test Manager could not delete the specified file.

**User Action:** Check the file protection and enter the command again.



ERREMOVALS, 'count' removal(s) completed with 'count' error(s)

**Explanation:** DEC/Test Manager encountered one or more errors while removing test descriptions or groups.

ERRINCOLL, error in collection 'name'

**Explanation:** The specified collection contains errors.

**User Action:** Use the VERIFY/RECOVER command to restore your library to a usable state.

ERRINSERTIONS, 'count' insertion(s) completed with 'count' error(s)

**Explanation:** DEC/Test Manager encountered one or more errors while inserting test descriptions or groups.

ERRINTD, error in test description 'name'

**Explanation:** The specified test description contains errors.

**User Action:** Use the VERIFY/RECOVER command to restore your library to a usable state.

ERRMODIFIES, 'count' modification(s) completed with 'count' error(s)

**Explanation:** DEC/Test Manager encountered one or more errors while modifying test descriptions or variables.

ERRORACT, error activating image 'filename'

**Explanation:** DEC/Test Manager could not activate the required shareable images.

**User Action:** See the system manager.

ERROVERRIDE, error associating new value of variable 'name' with test description 'name'

**Explanation:** DEC/Test Manager encountered one or more errors while associating an override value with the specified variable.

**User Action:** See the secondary message for more information.

ERRPAREXP, error parsing 'type' expression

**Explanation:** You entered a collection, group, test description, or variable expression with illegal syntax.

**User Action:** Correct the expression and enter the command again. See the secondary message for more information.

ERRSUBDIR, library subdirectory 'name' matches no existing collection

**Explanation:** A subdirectory exists in the DEC/Test Manager library that is not related to a valid collection.

**User Action:** Delete the subdirectory from the DEC/Test Manager library.

ERRUPDATES, 'count' update(s) completed with 'count' error(s)

**Explanation:** DEC/Test Manager encountered one or more errors while updating benchmark files.

ERRVERARC, archive list verified with errors

**Explanation:** DEC/Test Manager discovered an inconsistency or error in your DEC/Test Manager library that it cannot correct.

**User Action:** Restore the library from a backup tape to ensure that you are using a consistent library.

ERRVERCOL, collection list verified with errors

**Explanation:** DEC/Test Manager discovered an inconsistency or error in your DEC/Test Manager library that it cannot correct.

**User Action:** Restore the library from a backup tape to ensure that you are using a consistent library.

ERRVERCOLFIL, one or more files missing from collection subdirectory

**Explanation:** DEC/Test Manager discovered errors in the collection directory structure.

**User Action:** Re-create the collection.

ERRVERFRE, free space list verified with errors

**Explanation:** DEC/Test Manager discovered an inconsistency or error in your DEC/Test Manager library that it cannot correct.

**User Action:** Restore the library from a backup tape to ensure that you are using a consistent library.

ERRVERGRO, group list verified with errors

**Explanation:** DEC/Test Manager discovered an inconsistency or error in your DEC/Test Manager library that it cannot correct.

**User Action:** Restore the library from a backup tape to ensure that you are using a consistent library.

**ERRVERHEAD,** user header information verified with errors

**Explanation:** DEC/Test Manager discovered an inconsistency or error in your DEC/Test Manager library that it cannot correct.

**User Action:** Restore the library from a backup tape to ensure that you are using a consistent library.

**ERRVERSPACE,** contiguous space verified with errors

**Explanation:** DEC/Test Manager discovered an inconsistency or error in your DEC/Test Manager library that it cannot correct.

**User Action:** Restore the library from a backup tape to ensure that you are using a consistent library.

**ERRVERSTR,** string list verified with errors

**Explanation:** DEC/Test Manager discovered an inconsistency or error in your DEC/Test Manager library that it cannot correct.

**User Action:** Restore the library from a backup tape to ensure that you are using a consistent library.

**ERRVERTD,** test description list verified with errors

**Explanation:** DEC/Test Manager discovered an inconsistency or error in your DEC/Test Manager library that it cannot correct.

**User Action:** Restore the library from a backup tape to ensure that you are using a consistent library.

**ERRVERVAR,** variables list verified with errors

**Explanation:** DEC/Test Manager discovered an inconsistency or error in your DEC/Test Manager library that it cannot correct.

**User Action:** Restore the library from a backup tape to ensure that you are using a consistent library.

**EXIT,** leaving Review subsystem

**Explanation:** DEC/Test Manager exited the Review subsystem.

**EXITERR,** error on exit from Review subsystem

**Explanation:** DEC/Test Manager encountered one or more errors while exiting the Review subsystem.

EXPRIGNORED, 'expression type' expression ignored

**Explanation:** DEC/Test Manager ignored the specified expression because the command is syntactically incorrect. DEC/Test Manager will execute the command as if you had not entered the expression.

**User Action:** See the secondary message for more information.

EXTLOADFAIL, Connection made to node\_name, but failed to load Xtrap extension

**Explanation:** The connection was successfully made to the DECwindows workstation, but the server extension was not loaded by the DECwindows server.

**User Action:** Make sure that DECW\$SERVER\_EXTENSION\_XTRAP.EXE image is in SYS\$LIBRARY of the workstation that will be used by DEC/Test Manager. Restarting the DECwindows server on the workstation being used could also solve the problem assuming that the Xtrap extension is in SYS\$LIBRARY.

EXTRACTED, input file 'file-name' created

**Explanation:** DEC/Test Manager extracted the input file from the specified session file.

FAILTOCONN, client failed to connect

**Explanation:** A client that connected during a record operation did not connect to DEC/Test Manager during a play operation.

**User Action:** Determine why the client did not connect and try the operation again.

FILENOTEXIST, 'type' file 'name' does not exist

**Explanation:** The specified file does not exist. When a file (except the benchmark file) is named in a test description, the file must exist when a collection containing the test description is created.

**User Action:** See Chapter 3 to determine whether the missing file is one that you can create and modify.

FILMOVED, 'template/benchmark' file 'name' moved to CMS library

**Explanation:** DEC/Test Manager moved the specified benchmark or template file from your default directory to the specified VAX DEC/Code Management System (CMS) library.

FILNAMERR, error in 'type' file-name 'name'

**Explanation:** There is a syntax error in the specified file name.

**User Action:** Enter the command again with a proper file name. See the secondary message for more information.

FILNOTMOVED, 'template/benchmark' file 'name' not moved to CMS library 'library name'

**Explanation:** DEC/Test Manager was unable to move the specified file into the specified VAX DEC/Code Management System (CMS) library.

**User Action:** See the subsequent CMS messages to determine the problem. Then use CMS directly to place the file into the CMS library.

FILTERED, expression successfully filtered

**Explanation:** The expression was translated and the file was filtered.

FORCESYNCH, Forcing Synch on \*\*\* synch\_string \*\*\*

**Explanation:** A timeout on a marked synchronization record will force the DECwindows playback system to artificially match and satisfy the synchronization string.

**User Action:** Check that proper synchronization text records have been selected as synchronization points. The best records for selection should be guaranteed to occur during each playback of the session file.

FREEEFERR, unable to deallocate a system event flag

**Explanation:** A system service failed to deallocate an event flag.

**User Action:** Check quota and SYSGEN parameters. Consult with your system manager.

FROMSELF, cannot remove group 'name' from itself

**Explanation:** You tried to remove the specified group from itself. A group cannot be a member of itself, so you cannot remove a group from itself.

GETDVIERR, could not get information about device 'dev'

**Explanation:** A system service failed to obtain information about the specified device.

**User Action:** Verify that the device exists. Check the device protection and ownership.

GETEFERR, unable to allocate a system event flag

**Explanation:** A system service failed to allocate an event flag.

**User Action:** Check your quotas and SYSGEN parameters. Consult with your system manager.

HASFILES, directory 'name' contains files

**Explanation:** You tried to create a DEC/Test Manager library in a directory that is not empty.

**User Action:** Delete the files and reenter the CREATE LIBRARY command for the same directory or create the library in a different directory.

HASMEMBERS, group 'name' contains one or more groups or test descriptions

**Explanation:** You tried to delete a group that is not empty.

**User Action:** Remove any groups and test descriptions from the group, then enter the command again.

HASREFERENCE, variable 'name' referenced by a test description

**Explanation:** You tried to delete a variable that is referenced by one or more test descriptions.

**User Action:** Enter the SHOW VARIABLE/TEST\_DESCRIPTION command to list the tests that reference this variable. Then remove the references to the variable with the MODIFY TEST\_DESCRIPTION command and reenter the DELETE VARIABLE command.

HISNOTSTM, history file record format is not stream\_lf

**Explanation:** The library history file (00DTM.HIS) must be in stream\_lf (line feed) format. Note that if you edit the history file, the format may no longer be stream\_lf.

**User Action:** Use the VERIFY/RECOVER command to correct the file format.

HISTDEL, 'count' history records deleted

**Explanation:** DEC/Test Manager deleted some or all of your library history. By default, the deleted history information is placed in the file HISTORY.DMP in your default directory.

HOLDING, job 'job-name' (entry 'number') is holding in queue  
'queue-name'

**Explanation:** The specified job is waiting to execute.

IDENTICAL, files for test 'name' are identical

**Explanation:** The result and benchmark files for this result description are identical.

ILLCHAR, illegal character in 'name'

**Explanation:** The expression you entered contains a character that is not allowed in this context.

**User Action:** Correct the expression and enter the command again.

ILLEGALDEV, illegal device name specified

**Explanation:** You included an illegal device specification in a library directory specification or in a file specification.

**User Action:** Correct the device specification and enter the command again.

ILLFILEINLIB, 'type' file 'name' contains illegal DEC/Test Manager library specification

**Explanation:** The file specification cannot refer to the DEC/Test Manager library.

**User Action:** Correct the file specification and enter the command again.

ILLPRIORITY, /PRIORITY not in range 0 to 255

**Explanation:** You tried to submit a collection to the batch queue with a priority value outside the allowed range.

**User Action:** Correct the priority value and enter the command again.

ILLQUAL, 'name' is an illegal qualifier; use /TEST\_DESCRIPTION or /GROUP

**Explanation:** You cannot use the specified qualifier in a test-group-expression. You can use only the parameter qualifiers in this context, /GROUP and /TEST\_DESCRIPTION.

**User Action:** Verify the validity and position of the qualifier, then reenter the command.

ILLRECORD, unrecognized record type 'type'

**Explanation:** A record in the session file being processed begins with the specified unrecognized type designator.

**User Action:** Delete or modify the illegal record, or use a previous correct version of the file.

ILLTIME, /CPUTIME value is not a delta time

**Explanation:** You tried to submit a collection with a /CPUTIME qualifier value that is not a delta time.

**User Action:** Correct the value and enter the command again.

ILLWSDEFAULT, /WSDEFAULT not in range 1 to 65535

**Explanation:** You tried to submit a collection with a /WSDEFAULT qualifier value outside the allowed range.

**User Action:** Correct the value and enter the command again.

ILLWSEXTENT, /WSEXTENT not in range 1 to 65535

**Explanation:** You tried to submit a collection with a /WSEXTENT qualifier value outside the allowed range.

**User Action:** Correct the value and enter the command again.

ILLWSQUOTA, /WSQUOTA not in range 1 to 65535

**Explanation:** You tried to submit a collection with a /WSQUOTA qualifier value outside the allowed range.

**User Action:** Correct the value and enter the command again.



INPDEALLOC, DEallocated only ## of ## input buffers

**Explanation:** This is an internal DEC/Test Manager condition. It is not serious, and can occur during aborted DECwindows operations.

**User Action:** Submit a Software Performance Report (SPR).

INPERR, command line input error

**Explanation:** You entered a command containing a syntax error.

**User Action:** See the secondary messages for more information.

INPMBXINIT, Error Initializing input mailbox

**Explanation:** This is an internal DEC/Test Manager DECwindows condition.

**User Action:** Submit a Software Performance Report (SPR).

INPUTEOF, Unexpected end-of-file in input file

**Explanation:** The end of a DECwindows input file was reached before expected.

**User Action:** Check for input file corruption, and then submit a Software Performance Report (SPR).

INSERTED, 'object' 'name' inserted into group 'name'

**Explanation:** DEC/Test Manager inserted the group or test description into the specified group.

INSERTIONS, 'count' insertions completed

**Explanation:** DEC/Test Manager completed the specified number of insertions.

INTOSELF, cannot insert group 'name' into itself

**Explanation:** You tried to insert the specified group into itself. A group cannot be a member of itself.

INUSE, DEC/Test Manager library 'directory-spec' is in use, please wait

**Explanation:** The DEC/Test Manager library is currently being used by someone else. DEC/Test Manager will wait and automatically continue execution of your command as soon as the library is free.

INVCONTROLREC, Session file contains an invalid CONTROL record

**Explanation:** The DECwindows playback system found an invalid control record in the session file it was loading into memory.

**User Action:** Check for corrupted session file contents. This error should be caught by the extract or restore operations.

INVDECNETXPORT, Invalid transport Direction for DECnet

**Explanation:** This is an internal DEC/Test Manager DECwindows condition.

**User Action:** Submit a Software Performance Report (SPR).

INVDWOPT, invalid option specified for DECwindows test

**Explanation:** A command qualifier was selected which is invalid for DECwindows tests.

**User Action:** Remove the qualifier not allowed for DECwindows tests and reenter the command.

INVINTOPT, invalid option specified for interactive terminal test

**Explanation:** A command qualifier was selected which is invalid for interactive terminal tests.

**User Action:** Remove the qualifier not allowed for interactive terminal tests and reenter the command.

INVKBDCMD, Invalid keyboard command

**Explanation:** During DECwindows recording, any DEC/Test Manager command issued at the workstation that is not recognized produces this error and a bell.

INVKEYSYM, invalid keysym 'keysym'

**Explanation:** The keysym specified is unrecognized

**User Action:** See the Command Dictionary for the proper keysym format and reenter the command.

INVLINES, ## invalid lines were encountered in input file file\_spec

**Explanation:** When extracting or restoring a DECwindows session file or input file, this message indicates the number of invalid lines encountered.

**User Action:** Be sure to start with original recorded files, and be sure to fix all scripting errors identified by the restore command.

INVMBXPORT, Invalid transport Direction for MBX

**Explanation:** This is an internal DEC/Test Manager DECwindows condition.

**User Action:** Submit a Software Performance Report (SPR).

INVNONOPT, invalid option specified for noninteractive test

**Explanation:** A command qualifier was selected which is invalid for noninteractive tests.

**User Action:** Remove the qualifier not allowed for noninteractive tests and reenter the command.

INVPARMNUM, routine DTM\$DTM was called with an invalid number of parameters

**Explanation:** A call to the DEC/Test Manager callable interface has an incorrect number of parameters.

**User Action:** Correct the call (see Chapter 10).

INVPLAYRECORD, Playback encountered an invalid record in memory

**Explanation:** This is an internal DEC/Test Manager DECwindows condition.

**User Action:** Submit a Software Performance Report (SPR).

INVRECMODE, Invalid Script record mode

**Explanation:** This is an internal DEC/Test Manager DECwindows condition.

**User Action:** Submit a Software Performance Report (SPR) specifying this and any additional error messages that are seen.

INVRECORD, Invalid input record found at line ## of file file\_spec

**Explanation:** An Invalid DECwindows input record has been found. This record may be a session file record or an input file ASCII record. The extract or restore programs could not identify the record type.

**User Action:** The session file or the input file has become corrupted. Check past versions of the same files to track down where the problem began.

INVRECTYPE, Invalid Script record type

**Explanation:** This is an internal DEC/Test Manager DECwindows condition.

**User Action:** Submit a Software Performance Report (SPR) specifying this and any additional error messages that are seen.

INVRESOURCEMSK, Invalid resource bit mask in clean\_up() switch

**Explanation:** This is an internal DEC/Test Manager DECwindows condition.

**User Action:** Submit a Software Performance Report (SPR).

INVSTRDES, invalid string descriptor at virtual address 'address'

**Explanation:** A string descriptor containing the command line passed to the DEC/Test Manager callable interface has an invalid format.

**User Action:** Correct the string descriptor format.

INVSYNTAX, Invalid script syntax at line ##

**Explanation:** The specified line contains a scripting error.

**User Action:** Check the specified source lines for typographical errors, errors specifying a script record, or otherwise corrupted record text.

INVTRMCHR, invalid termination character 'string' specified

**Explanation:** The string you specified for the /TERMINATION\_CHARACTER qualifier is invalid.

**User Action:** Correct the string specification and enter the command again.

INVXPORVAL, Invalid dcb w\_transport value.

**Explanation:** This is an internal DEC/Test Manager DECwindows condition.

**User Action:** Submit a Software Performance Report (SPR).

INVXTRAPEXT, Invalid Xtrap Extension: OS Ident = #, Owner = #,  
Version = #

**Explanation:** DEC/Test Manager has identified a version mismatch between itself and the Xtrap extension currently loaded on the workstation.

**User Action:** Make sure that the latest DECW\$SERVER\_EXTENSION\_XTRAP.EXE has been placed in SYS\$LIBRARY of the workstation in question. Restart the DECwindows server on the workstation in question to load the correct Xtrap extension.

IOERROR, error 'opening' file 'name'

**Explanation:** An I/O error occurred while DEC/Test Manager was either reading, writing, opening, or closing the specified file. The type of file, the specified action, and the file name are included in the message text.

**User Action:** Verify that the file exists and check the file protection and ownership. See the secondary messages for more information.

ISMEMBER, 'object' 'name' is a member of group 'name'

**Explanation:** You tried to delete a group or test description that belongs to another group.

**User Action:** Remove the test description or group from all groups to which it belongs, then enter the command again.

LF\_FILLS, recording terminal needs no lffill, display terminal does

**Explanation:** The terminal on which you are monitoring an executing test has characteristics different from the terminal on which the test was recorded.

**User Action:** DEC/Test Manager is executing the test correctly (even though the display may look incorrect) and will create a proper result file. Use the Review subsystem SHOW/RESULT command to display the result file.

LIBIS, DEC/Test Manager library is 'directory-spec'

**Explanation:** Your DEC/Test Manager library is now defined to be the specified directory.

LINEOFERR, At line ## in file file\_spec

**Explanation:** This message indicates the line number of an error found when restoring a DECwindows input file.

**User Action:** Locate and correct the specified error.

LOSTDECNETLINK, Lost DECnet link. Stopping Device

**Explanation:** This indicates that DEC/Test Manager has lost its connection to the workstation being used as a result of a problem with the workstation DECnet link.

**User Action:** Try to identify the problem that caused the workstation to abort the DECnet link to DTM. If recording or playback was active through the quitting of a session, this error can be expected. Quitting from a session shuts down all DEC/Test Manager communication with the remote workstation.

LOWERCAS, recording terminal had lowercase, display terminal does not

**Explanation:** The terminal on which you are monitoring an executing test has characteristics different from the terminal on which the test was recorded.

**User Action:** DEC/Test Manager is executing the test correctly (even though the display may look incorrect) and will create a proper result file. Use the Review subsystem SHOW/RESULT command to display the result file.

MASKCANTALLOC, 'type' block 'name'

**Explanation:** DEC/Test Manager cannot allocate enough internal memory for editing this file.

**User Action:** Increase virtual memory quotas.

**MASKCANT\_GET\_ATTRS, 'type' block 'name'**

**Explanation:** DEC/Test Manager cannot obtain the frame attributes for a particular frame in this file. As a result DEC/Test Manager cannot determine whether or not masked regions already exist for this frame.

**User Action:** Check protections on the file. User should have at least read access.

**MASKDDIFNULL, 'type' block 'name',**

**Explanation:** The file is empty. It does not contain any image frames.

**MAXTASKS, cannot create task, reached task limit**

**Explanation:** An attempt was made to create more tasks than DEC/Test Manager allows.

**User Action:** Wait for an existing task to terminate and try the operation again.

**MAXWARN, Maximum number (##) of syntax warnings reached**

**Explanation:** The message indicates that the maximum number of errors has been reached processing a DECwindows input file.

**User Action:** Correct the specified errors and perform the extract or restore again.

**MECHFORM, recording terminal supports FF, but display terminal does not**

**Explanation:** The terminal on which you are monitoring an executing test has characteristics different from the terminal on which the test was recorded.

**User Action:** DEC/Test Manager is executing the test correctly (even though the display may look incorrect) and will create a proper result file. Use the Review subsystem SHOW/RESULT command to display the result file.

MECH\_TAB, recording terminal supported tabs, display terminal does not

**Explanation:** The terminal on which you are monitoring an executing test has characteristics different from the terminal on which the test was recorded.

**User Action:** DEC/Test Manager is executing the test correctly (even though the display may look incorrect) and will create a proper result file. Use the Review subsystem SHOW/RESULT command to display the result file.

MEMALLOCFAIL, Unable to allocate memory for data\_structure

**Explanation:** This is an internal DEC/Test Manager DECwindow condition.

**User Action:** Check your system's memory configuration for tuning problems. Submit a Software Performance Report (SPR).

MISBLKSTR, a 'type' block was not hit during pass 1

**Explanation:** DEC/Test Manager discovered an inconsistency or error in your DEC/Test Manager library that it cannot correct.

**User Action:** Restore the library from a backup tape to ensure that you are using a consistent library.

MODIFICATIONS, 'count' modification(s) completed

**Explanation:** DEC/Test Manager modified the specified number of test descriptions or variables.

MODIFIED, 'object' 'name' modified

**Explanation:** DEC/Test Manager modified the specified test description or variable.

MRKFORINSERT, test\_description 'name' marked for insertion

**Explanation:** DEC/Test Manager marked the specified test description for insertion into the group created when you exited the Review subsystem.



MSSBLKSTR, there were 'count' 'identifier' type blocks found on pass 1,  
there were 'count' blocks found on pass 2

**Explanation:** DEC/Test Manager discovered an inconsistency or error in your DEC/Test Manager library that it cannot correct.

**User Action:** Restore the library from a backup tape to ensure that you are using a consistent library.

MUSTBEDIR, 'string' must be a directory specification

**Explanation:** DEC/Test Manager expected a directory specification where you entered the specified string of characters.

**User Action:** Correct the parameter and enter the command again.

MUSTBEGLOB, variable 'name' must be global for CREATE  
COLLECTION to override

**Explanation:** The variable you specified on the CREATE COLLECTION command is a local variable.

**User Action:** Verify that you entered the correct variable name, then reenter the command with a variable that is global in scope.

NETMBXREAD, Unexpected IOSB Error on network command MBX read

**Explanation:** This is an internal DEC/Test Manager DECwindows condition.

**User Action:** Submit a Software Performance Report (SPR).

NETMSGBADPARAM, Process\_netcmd() encountered an Invalid link  
message code

**Explanation:** This is an internal DEC/Test Manager DECwindows condition.

**User Action:** Submit a Software Performance Report (SPR).

NETREADFAILED, Process\_netcmd() failed: could not queue read to  
netcmd MBX

**Explanation:** This is an internal DEC/Test Manager DECwindows condition.

**User Action:** Submit a Software Performance Report (SPR).

NEWASSOCVAL, superseding old association value between variable  
'name' and test description 'name'

**Explanation:** This variable and test description are already associated. The new value you specified supersedes the existing value.

NEWDEF, 'file' is the new default collection 'prologue/epilogue'

**Explanation:** You specified a new default collection prologue file or collection epilogue file.

NEWWIDTH, recording terminal has width n, display terminal has  
width m

**Explanation:** The terminal on which you are monitoring an executing test has characteristics different from the terminal on which the test was recorded.

**User Action:** DEC/Test Manager is executing the test correctly (even though the display may look incorrect) and will create a proper result file. Use the Review subsystem SHOW/RESULT command to display the result file.

NEW\_TERM, recording terminal was a 'terminal-type', display terminal  
is a 'terminal-type'

**Explanation:** The terminal on which you are monitoring an executing test has characteristics different from the terminal on which the test was recorded.

**User Action:** DEC/Test Manager is executing the test correctly (even though the display may look incorrect) and will create a proper result file. Use the Review subsystem SHOW/RESULT command to display the result file.

NOASSOC, variable 'name' was not associated with test-description  
'name'

**Explanation:** You attempted to remove a nonexistent association between the specified variable and test description.

NOATTACH, could not attach to process

**Explanation:** DEC/Test Manager could not issue the specified ATTACH command.

**User Action:** Verify the existence of the process to which you want to attach and verify that it is part of your job. See the secondary messages for more information.

NOCANCEL, Error cancelling device

**Explanation:** This is an internal DEC/Test Manager DECwindows condition.

**User Action:** Submit a Software Performance Report (SPR).

NOCMDKEYCODE, KEYSYM to KEYCODE translation for command key failed

**Explanation:** This is an internal DEC/Test Manager DECwindows condition.

**User Action:** Submit a Software Performance Report (SPR).

NOCOMPARE, collection 'name' not compared

**Explanation:** DEC/Test Manager could not compare the specified collection.

NOCONVERT, error converting your 'V1' library to 'V2'

**Explanation:** DEC/Test Manager could not convert your library.

**User Action:** See the secondary message for more information.

NOCOPY, error copying 'object' 'name'

**Explanation:** DEC/Test Manager could not copy the specified collection, group, test description, or variable.

**User Action:** See the secondary message for more information.

NOCREATE, error creating 'object' 'name'

**Explanation:** DEC/Test Manager could not create the specified collection, group, test description, variable, or library.

NOCTRLC, Control-C could not be enabled on terminal device

**Explanation:** This is an internal DEC/Test Manager DECwindow condition indicating that control-c will not be recognized from the current terminal.

**User Action:** Submit a Software Performance Report (SPR) specifying this and any additional error messages that are seen.

NOCTRLY, Control-Y could not be enabled on terminal device

**Explanation:** This is an internal DEC/Test Manager DECwindow condition indicating that control-y will not be recognized from the current terminal.

**User Action:** Submit a Software Performance Report (SPR) specifying this and any additional error messages that are seen.

NOCURRE, you are not currently positioned at a result description

**Explanation:** You entered a Review subsystem command that requires you to be positioned at a result description.

**User Action:** Enter a Review subsystem command to select a result description, then enter the command again.

NODASSGN, Error Deassigning device

**Explanation:** This is an internal DEC/Test Manager DECwindow condition.

**User Action:** Submit a Software Performance Report (SPR).

NODECUI, DECwindows files cannot be displayed on a terminal

**Explanation:** You tried to display a DECwindows result or benchmark file on a terminal.

**User Action:** Use a DECwindows workstation to display the file.

NODECWFMT, 'library-name' file 'file-name' is not in DECwindows format

**Explanation:** When attempting to compare a DECwindows test, DEC/Test Manager found files that were not run in the DECwindows test format.

**User Action:** Rerun the test to create new files for comparison.

NODECWINDOWS, DECwindows is not installed on this system

**Explanation:** DECwindows is not installed on this system.

**User Action:** See the system manager.

NODEFINE, your DEFINE/KEY command is syntactically incorrect

**Explanation:** An error exists in the DEFINE/KEY command you entered.

**User Action:** See the secondary messages for more information.

NODEL, cannot delete 'object'

**Explanation:** DEC/Test Manager could not delete the specified collection, group, test description, or variable.

**User Action:** See the secondary message for more information.

NODELETE, error deleting 'object' 'name'

**Explanation:** DEC/Test Manager could not delete the specified collection, group, test description, or variable.

**User Action:** See the secondary message for more information.

NODELETIONS, no 'name' deletions performed

**Explanation:** DEC/Test Manager could not delete the specified items.

**User Action:** See the secondary message for more information.

NODELFUTURE, cannot delete history of future events

**Explanation:** You specified a time that is later than the current time.

**User Action:** Correct the time value and enter the command again.

NODISPLAY, could not display benchmark for test 'name'

**Explanation:** The DISPLAY/BENCHMARK command could not display the specified benchmark file.

**User Action:** See the secondary messages for more information.

NOEXTRACT, input file not extracted from 'session file'

**Explanation:** DEC/Test Manager could not extract an input file from the specified session file.

**User Action:** See the secondary message for more information.

NOHIS, no history records found

**Explanation:** The current DEC/Test Manager library contains no history records for the specified object.

NOINPUTFILE, Could not open ASCII Input file file\_spec

**Explanation:** The ASCII DECwindows input file could not be opened.

**User Action:** Note the error messages associated with this error.

NOINSERT, error inserting 'object' 'name' into group 'name'

**Explanation:** DEC/Test Manager could not insert the test description or group into the specified group.

**User Action:** See the secondary message for more information.

NOINSERTIONS, no 'type' insertions performed

**Explanation:** DEC/Test Manager did not perform the specified insertions.

**User Action:** See the secondary message for more information.

NOLCKKEYCODE, KEYSYM to KEYCODE translation for command lock key failed

**Explanation:** This is an internal DEC/Test Manager DECwindow condition.

**User Action:** Submit a Software Performance Report (SPR).

NOMATCH, no match was found for 'type' expression 'expression'

**Explanation:** DEC/Test Manager could not find any item in the library or collection to match your command line expression.

**User Action:** Verify the spelling and meaning of your expression, then reenter the command.

NOMARK, cannot mark result description 'name' for insertion

**Explanation:** DEC/Test Manager could not mark the named result description for insertion upon leaving the Review subsystem.

**User Action:** See secondary message for more information.

NOMEM, Not enough internal memory to contain session file file\_spec

**Explanation:** The DECwindows playback system ran out of memory loading a session file. The session file was too big to fit within the memory constraints of the playback system, which is currently 1000 blocks.

**User Action:** Subdivide your script into smaller tasks, and submit a Software Performance Report (SPR).

NOMODARG, arguments do not specify any modifications to 'object'

**Explanation:** DEC/Test Manager did not modify the test description or variable because you did not specify any fields to be modified.

**User Action:** Enter the command again with one or more qualifiers specifying fields to modify.

NOMODIFIES, no 'type' modifications performed

**Explanation:** DEC/Test Manager performed no modifications.

**User Action:** See the secondary message for more information.

NOMODIFY, error modifying 'object' 'name'

**Explanation:** DEC/Test Manager could not modify the specified test description or variable.

**User Action:** See the secondary message for more information.

NOMORE, no more 'type' result descriptions found

**Explanation:** DEC/Test Manager could not find a result description matching the result description expression.

NOMOVE, result description position not changed

**Explanation:** DEC/Test Manager did not move you to the specified result description because the command you entered was syntactically incorrect.

**User Action:** See the secondary message for more information.

NONETCMD, Allocation of the network link command buffer failed

**Explanation:** This is an internal DEC/Test Manager DECwindow condition.

**User Action:** Submit a Software Performance Report (SPR) specifying this and any additional error messages that are seen.

NONETOBJ, Network object has not been declared

**Explanation:** This is an internal DEC/Test Manager DECwindow condition.

**User Action:** Submit a Software Performance Report (SPR) specifying this and any additional error messages that are seen.

NONTTYRECORDER, recording may only be performed when running on a terminal

**Explanation:** You cannot record a terminal session while executing in batch.

**User Action:** Verify that SYS\$INPUT and SYS\$OUTPUT are specified as terminals.

NONULL, NULL not found in Record ###

**Explanation:** DEC/Test Manager Restore of an input file expects that records be null terminated. A NULL terminator was not found.

**User Action:** Check for input file corruption, and then submit a Software Performance Report (SPR).

NOPARTWILD, no partial wildcards allowed in result description expression

**Explanation:** Partial wildcards are not allowed as parameters to the SELECT Review subsystem command.

**User Action:** Enter the command again specifying a result description name, the asterisk (\*) wildcard, or no parameter.

NOPRINT, cannot print 'file type' 'name'

**Explanation:** DEC/Test Manager could not print the specified file

**User Action:** See the secondary message for more information.



NOPRINTQD, error submitting job to print queue

**Explanation:** DEC/Test Manager could not submit the specified files to the print field queue.

**User Action:** See the secondary message for more information.

NORECLAIM, error reclaiming loose type 'type' block

**Explanation:** The DEC/Test Manager VERIFY/REPAIR command found a loose block of the specified type in the library but was unable to move it to the proper location.

**User Action:** Restore your library from a backup tape.

NORECORD, No output record produced - program continuing

**Explanation:** When DEC/Test Manager is extracting a DECwindows session file, any invalid binary record will produce this error.

**User Action:** Check that a previous restore has not corrupted the session file being extracted. Use the original session file. Re-record the test.

NORECOVER, error recovering library

**Explanation:** DEC/Test Manager could not recover your library.

**User Action:** Restore your library from a backup tape.

NORECREATE, error re-creating collection 'name'

**Explanation:** DEC/Test Manager could not re-create your collection.

NOREF, unable to reference 'directory'

**Explanation:** You tried to execute a DEC/Test Manager command without first selecting a DEC/Test Manager library.

**User Action:** Use the SET LIBRARY command to select a DEC/Test Manager library, then enter the command again.

NOREMARK, error adding remark to DTM library

**Explanation:** DEC/Test Manager could not enter your remark in the library history file.

**User Action:** See the secondary messages for more information.

NOREMOVAL, error removing 'object' 'name' from group 'name'

**Explanation:** DEC/Test Manager could not remove the group or test description from the specified group.

**User Action:** See the secondary message for more information.

NOREMOVALS, no 'type' removals performed

**Explanation:** DEC/Test Manager performed no removals.

NOREPAIR, error repairing library

**Explanation:** DEC/Test Manager could not repair your library.

**User Action:** Restore your library from a backup tape.

NORESTORE, session file not restored from 'file-name'

**Explanation:** DEC/Test Manager could not restore the session file from the specified input file.

**User Action:** See the secondary message for more information.

NORESUBMIT, collection 'name' cannot be resubmitted

**Explanation:** DEC/Test Manager could not resubmit the specified collection.

**User Action:** See the secondary message for more information.

NORETRIEVE, could not retrieve 'information' for test 'name'

**Explanation:** DEC/Test Manager could not access the specified information for the specified test description.

NOREVIEW, error reviewing collection

**Explanation:** DEC/Test Manager could not review the specified collection.

**User Action:** See the secondary message for more information.

NOSESSIONFILE, Could not open session file file\_spec

**Explanation:** The session file could not be opened.

**User Action:** Note the error messages associated with this error.

NOSET, could not set default 'item'

**Explanation:** DEC/Test Manager could not establish the collection prologue file, collection epilogue file, benchmark directory, or template directory.

**User Action:** Correct the expression and reenter command.

NOSHOW, error showing 'object' 'name'

**Explanation:** DEC/Test Manager could not display the specified result, benchmark, or differences file; or it could not display the test description, variable, group, or collection.

NOSINCE, error executing /SINCE operation

**Explanation:** DEC/Test Manager could not display the library history as specified by the /SINCE qualifier.

**User Action:** See the secondary messages for more information. (see HISNOTSTM)

NOSPAWN, could not spawn a subprocess

**Explanation:** DEC/Test Manager could not spawn a subprocess.

**User Action:** Verify your process quotas and job limits, then reenter the command.

NOSPEC, no 'item' specified

**Explanation:** No default benchmark directory, template directory, collection prologue file, or collection epilogue file exists.

NOSRCHLST, search lists are not allowed in this context: 'name'

**Explanation:** You cannot use a search list here.

NOSTOP, collection 'name' was not stopped

**Explanation:** DEC/Test Manager did not stop the collection.

NOSUBMIT, collection 'name' cannot be submitted

**Explanation:** DEC/Test Manager could not submit the specified collection.

**User Action:** See the secondary message for more information.

NOSUCHJOB, job 'number' does not exist in queue 'name'

**Explanation:** DEC/Test Manager could not find an executing batch job for the collection.

**User Action:** Use the VERIFY/RECOVER command to restore your library to a usable state, then enter the command again.

NOSYNCHRECORD, Can't force synch: No entry to remove from synch\_ queue

**Explanation:** This is an internal DEC/Test Manager DECwindow condition.

**User Action:** Submit a Software Performance Report (SPR).

NOTAPPLIC, the /STRING and /NUMERIC qualifiers are not applicable to logical variable 'name'

**Explanation:** DEC/Test Manager did not modify the variable type because the variable currently has a logical type and therefore cannot have the string or numeric type.

**User Action:** Verify that the current usage for the variable is correct.

NOTCHANGED, the default 'name' directory was not changed

**Explanation:** DEC/Test Manager could not change the specified directory.

NOTCRELIB, first history record is not DTM CREATE LIBRARY transaction

**Explanation:** The first record of every history file should be a transaction of the CREATE LIBRARY command. It is likely that the history file has been edited.

NOTDELTATIME, DTM\$DELAY\_TIMEOUT must specify a DELTA TIME -DELTA Times are of the form '0 00:00:00.0'. -Using default timeout of 0 00:03:00.00>

**Explanation:** The DEC/Test Manager timeout logical has been incorrectly specified. The DEC/Test Manager default timeout value will be used.

**User Action:** Redefine the logical with a valid VMS DELTA time specification as noted.

NOTDTMLIB, 'directory\_spec' is not a valid DEC/Test Manager library

**Explanation:** You specified a directory that is not a valid DEC /Test Manager library.

**User Action:** Correct the directory specification and enter the command again.

NOTESTCMP, no tests in collection 'name' could be compared

**Explanation:** No tests ran in this collection; consequently, no result files were produced.

**User Action:** This collection can be resubmitted or re-created.

NOTFILTERED, expression 'expression' could not be filtered

**Explanation:** There was a problem either translating the expression or finding the resolved file specification.

**User Action:** See the secondary message for more information.

NOTFINISHED, collection 'name' has not finished running

**Explanation:** You tried to compare a collection that is still executing.

**User Action:** Enter the command again when the collection has finished executing.

NOTFOUND, 'object' not found

**Explanation:** DEC/Test Manager could not find the specified collection, group, test description, or variable.

NOTINTER, test description 'name' is not interactive

**Explanation:** You performed an operation reserved for interactive tests on a noninteractive test.

NOTINTEST, no 'field' associated with this test description

**Explanation:** The prologue, epilogue, variable, or filter is not associated with this test.

**User Action:** Enter the command again without the qualifier.

NOTLATIN1, the specified character is not in the Latin 1 keysym encodings

**Explanation:** The termination character specified for a DECwindows test cannot be found in the DECwindows Latin1 keysym encodings.

**User Action:** Enter the command again with a valid termination character.

NOTMASKED, 'type' block 'name'

**Explanation:** The DDIF file was not masked because an error was encountered.

**User Action:** See secondary message.

NOTMEMBER, 'type' 'name' is not currently a member of group 'name'

**Explanation:** You selected a group or test description for removal from a specified group, but the test description or group is not currently a member of the specified group.

NOTTRANSLATE, unrecognized sequence 'sequence' will not be translated

**Explanation:** DEC/Test Manager cannot provide a translation for a special string or for a recording function or nonprinting text while creating the input or session file. DEC/Test Manager will copy the untranslated sequence to the file.

**User Action:** Verify that the sequence is correct. If it is correct, use a text editor to include the correct translation in the file being created.

NOTRUN, collection 'name' was not found running in the batch queues

**Explanation:** You tried to stop a collection that is not currently executing in any batch queue.

NOTYETRAN, collection 'name' has not yet been run

**Explanation:** You tried to review or compare a collection that has not yet been executed.

**User Action:** Use the DTM RUN or SUBMIT command to execute the collection, then enter the command again.

NOUPDATE, the benchmark file for result description 'name' has not been updated

**Explanation:** DEC/Test Manager encountered an error while updating the benchmark file.

**User Action:** See the secondary message for more information.

NOVERIFY, DEC/Test Manager library 'name' not verified

**Explanation:** DEC/Test Manager encountered errors that it cannot correct while trying to verify the library.

**User Action:** Restore the library from a backup tape to ensure that you are using a consistent library.

NOWLDCARD, wildcards are not allowed in this context: 'string'

**Explanation:** You specified a wildcard character where none is allowed.

**User Action:** Replace the expression with a name and enter the command again.

NO\_PC\_DEV, your system does not contain the PC: device

**Explanation:** The device PC: does not exist on your system.

**User Action:** Verify that the PCDRIVER is installed on your system and check the system startup procedure to ensure that the PCDRIVER is being loaded.

NULLEXPR, a null 'type' expression is not allowed in this context

**Explanation:** You entered an empty string for a required parameter.

**User Action:** Enter the command again with a valid parameter.

NULLNAME, a null value for qualifier 'name' is not permitted in this context

**Explanation:** You entered an empty string for a required qualifier.

**User Action:** Enter the command again with a valid qualifier.

NULLNUM, numeric symbol variables cannot have a null value

**Explanation:** Numeric symbols with null values generate invalid DCL symbol assignment statements of the form VARIABLE =. This is not allowed.

**User Action:** If you specified both the /VALUE and the /NUMERIC qualifiers with a MODIFY or CREATE command, enter the command again with one qualifier or the other but not both. If you were attempting to modify an existing variable, examine it with the SHOW VARIABLE command to verify that your modification is sensible.

NULLOG, logical variables cannot have a null value

**Explanation:** Logicals with null values generate invalid DCL DEFINE statements.

**User Action:** If you specified both the /VALUE and the /LOGICAL qualifiers with a MODIFY or CREATE command, enter the command again with one qualifier or the other but not both. If you were attempting to modify an existing variable, examine it with the SHOW VARIABLE command to verify that your modification is sensible.

OLDLIB, your current library is an old version, please use the CONVERT command

**Explanation:** Your current library was created by a previous incompatible version of DEC/Test Manager.

**User Action:** See Chapter 7 for instructions for converting your library.

OPENIN, error opening 'name' as input

**Explanation:** The COMPARE command could not open the specified file for input.

OPENOUT, error opening 'name' for output

**Explanation:** The COMPARE command could not open the specified file for output.

ORIGBENMISS, original benchmark 'name' no longer exists; current benchmark is 'name'

**Explanation:** The original benchmark file has been changed.



OUTDEALLOC, DEallocated only ## of ## output buffers

**Explanation:** This is an internal DEC/Test Manager condition. It is not serious, and can occur during aborted DECwindows operations.

**User Action:** Submit a Software Performance Report (SPR).

OUTFILEINCOMP, Output file file\_spec is incomplete

**Explanation:** This indicates that errors have been found while processing a DECwindows input file. The output file has been generated but is not complete and should not be used.

**User Action:** Correct the specified errors and perform the restore again.

OUTMBXINIT, Error Initializing output mailbox

**Explanation:** This is an internal DEC/Test Manager DECwindows condition.

**User Action:** Submit a Software Performance Report (SPR).

OVERRUN, data overrun using PC: device

**Explanation:** Your program transmitted data at a rate that DEC/Test Manager could not handle.

**User Action:** Verify that your parent process is not running at a low priority, or rerun your program when the system is not as heavily loaded.

PAGESIZE, recording terminal has PAGESIZE *n*, display terminal PAGESIZE *m*

**Explanation:** The terminal on which you are monitoring an executing test has characteristics different from the terminal on which the test was recorded.

**User Action:** DEC/Test Manager is executing the test correctly (even though the display may look incorrect) and will create a proper result file. Use the Review subsystem SHOW/RESULT command to display the result file.

PARTCMP, collection 'name' was partially compared

**Explanation:** DEC/Test Manager partially compared the specified collection.

PC\_CHECK, check that PCDRIVER is installed and PC0: and PTY0: exist

**Explanation:** PCDRIVER is not present or is improperly installed.

**User Action:** Check the system startup procedure to ensure that PCDRIVER is loaded. Verify that the devices PC0:, PTY0:, and VTA0: exist.

PC\_READ, error reading from PC: device

**Explanation:** An error occurred while reading from the PC: device.

**User Action:** Check quotas and protections.

PC\_WRITE, error writing to PC: device

**Explanation:** An error occurred while writing to the PC: device.

**User Action:** Check default protections and process quotas.

PENDING, job 'job-name' (entry 'number') is pending in queue  
'queue-name'

**Explanation:** The job is waiting for room in the batch queue before beginning its execution.

PKT\_NOTFILTERED, string not filtered, unrecognized packet format

**Explanation:** DEC/Test Manager could not filter a string in a DECwindows result or benchmark file.

**User Action:** Regenerate the result or benchmark file.

PLAYFAILED, play of session 'name' has failed

**Explanation:** DEC/Test Manager could not execute the specified test.

**User Action:** See the secondary messages for more information.

PRINT, file 'name' of test 'name' selected for printing

**Explanation:** DEC/Test Manager marked the specified file for printing.

PRINTQD, print job has been sent to the print queue

**Explanation:** DEC/Test Manager submitted the specified job to the print queue.

PROCEEDING, proceeding with command execution

**Explanation:** Your library is now free and DEC/Test Manager is continuing with execution of your command.

PTY\_ERR, could not create pseudoterminal for 'name'

**Explanation:** DEC/Test Manager could not create a PTY: device like the specified device.

**User Action:** Check the SYSGEN parameters and process quotas, then verify that PCDRIVER is loaded. See the secondary messages for more information.

QUALEXPRCONFLICT, the '/' qualifier' conflicts with the 'expression type' expression

**Explanation:** The specified qualifier and expression cannot both be included on the same command.

**User Action:** See the primary message for more information.

READOUTPUTERR, Read\_workstation\_output() FAILED with VMS status ##

**Explanation:** This is an internal DEC/Test Manager DECwindows condition.

**User Action:** Submit a Software Performance Report (SPR).

REASSURE, nonetheless, result file will be properly built

**Explanation:** Previous informational messages stated that the terminal on which you are monitoring an executing test has characteristics different from the terminal on which the test was recorded. This message assures you that DEC/Test Manager is executing the test correctly (even though the display may look incorrect) and will create a proper result file.

RECORDING, Recording...

**Explanation:** This indicates that DECwindows recording has successfully begun.

RECORDEND, Recording stopped

**Explanation:** DECwindows recording has been successfully stopped.

RECGRO, inserting group 'name' into group 'name' would create a recursive group

**Explanation:** You cannot build a recursive group, a group that contains itself.

**User Action:** Create a new group with a different name and insert the group into that group.

RECLAIMED, loose type 'type' block reclaimed

**Explanation:** The DTM VERIFY/REPAIR command has identified a loose block in the library and moved it back to its proper place.

RECNOTNEC, recovery is not necessary; DEC/Test Manager library 'directory-spec' is in a safe state

**Explanation:** The DEC/Test Manager library is not in need of recovery. DEC/Test Manager made no changes to the library.

RECORDED, test 'name' has been successfully recorded

**Explanation:** DEC/Test Manager recorded the interactive session.

RECORDFAILED, session file for test description 'name' has not been recorded

**Explanation:** DEC/Test Manager could not record the interactive session.

**User Action:** See the secondary messages for more information.

RECOVERED, DEC/Test Manager library 'directory-spec' recovered

**Explanation:** Your DEC/Test Manager library has been recovered and is ready for use.

RECREATED, collection 'name' has been re-created

**Explanation:** DEC/Test Manager deleted and re-created the specified collection.

REMARK, remark added to history file

**Explanation:** DEC/Test Manager added your remark to the history file.

REMOVALS, 'count' removals completed

**Explanation:** DEC/Test Manager performed the specified number of removals.

REMOVED, 'object' 'name' removed from group 'name'

**Explanation:** DEC/Test Manager removed the group or test description from the specified group.

REMQFAILURE, Unable to REMQ a data\_structure

**Explanation:** This is an internal DEC/Test Manager DECwindows condition.

**User Action:** Submit a Software Performance Report (SPR).

REPAIRED, DEC/Test Manager library 'directory-spec' repaired

**Explanation:** Your DEC/Test Manager library has been repaired and is ready for use.

REQUIRESYNCH, The Record at line ## SHOULD be a SYNCH Record.  
(It matches Synch Record \*\*\* string \*\*\* at line ##)

**Explanation:** The specified line number contains a text record that precedes an identical text record that was marked as a synchronization point.

**User Action:** Make the record at the specified line a synchronization record, or choose another synchronization text record instead of the currently marked synchronization point. A Synchronization record must be unique back to but not including the last synchronization record.

RESDESCERR, the information stored in this result description is in error

**Explanation:** One of the files associated with the result description is in error or has been deleted.

**User Action:** You cannot review this result description as it is. See Chapter 5 for information on using the REVIEW INSERT command and on reexecuting the test.

RESERVNAM, 'name' is reserved for DEC/Test Manager use only

**Explanation:** The name you specified is reserved for use only by DEC/Test Manager.

**User Action:** Enter the command again with a different name.

RESTORED, session file 'file-name' created

**Explanation:** DEC/Test Manager restored the session file from the specified input file.

RESTRMERR, could not restore terminal characteristics for 'terminal'

**Explanation:** A system service failed to restore your terminal's characteristics.

**User Action:** See the secondary messages for more information.

RESUBMITTED, collection 'name' has been resubmitted

**Explanation:** The collection is now executing in batch.

RETTODTM, Returning to DTM

**Explanation:** This is an internal DEC/Test Manager DECwindows condition indicating that a signaled error is returning control to the DEC/Test Manager command level.

**User Action:** Submit a Software Performance Report (SPR) specifying this and any additional error messages that are seen.

RUNFAILED, run of collection 'name' has failed

**Explanation:** DEC/Test Manager could not execute the specified collection.

**User Action:** See the secondary messages for more information.

SAVSCREEN, Saving Screen...

**Explanation:** During a DECwindows recording session, this message indicates that DEC/Test Manager is saving the pixels from the workstation display screen.

**User Action:** The user should not use the workstation keyboard or pointer until the save screen is complete. The completion of the save screen operation is signaled by a bell at the workstation and a message to sys\$output if sys\$output is active.

SAVSCRNCMPLT, Save screen complete

**Explanation:** The pixels of a workstation display screen have been saved to a file.

**User Action:** Recording may continue.

SAVSCR\_CDDIF, Error creating ddif file to receive screens

**Explanation:** An error was encountered trying to open the DDIF file that holds the pixels of a saved DECwindows display screen.

**User Action:** Try to identify why the file was unable to be created.

SAVSCR\_NOMEM, Insufficient virtual memory to save screen

**Explanation:** There was not enough system memory to fully process the saving of a DECwindows save screen during record.

**User Action:** Tune system memory usage using the Monitor Pool DCL command and other system management resources to identify any system problems.

SCOPETRM, recording terminal was a scope, but display terminal is not

**Explanation:** The terminal on which you are monitoring an executing test has characteristics different from the terminal on which the test was recorded.

**User Action:** DEC/Test Manager is executing the test correctly (even though the display may look incorrect) and will create a proper result file. Use the Review subsystem SHOW/RESULT command to display the result file.

SENSEMODEERR, could not get terminal characteristics for 'terminal'

**Explanation:** A system service failed to obtain the specified terminal's device characteristics.

**User Action:** Verify that the terminal exists and check the terminal's protection and ownership.

SESSION\_BAD, session file 'name' has invalid format

**Explanation:** The session file contains invalid data. Most likely, you edited a session file created by DEC/Test Manager or you created a session file without using the appropriate DEC/Test Manager commands.

**User Action:** If you modified the session file, verify your changes, or replace the session file with a backup version. Make sure that the first record in the session file is a 12-byte terminal characteristics block. See Chapter 8 for more information.

SESSIONFILELOAD, # script lines processed, # blocks read

**Explanation:** This is an informational message indicating the number of session file lines that were read into memory and the corresponding block count that the lines represent.

SESSION\_READ, error reading session file 'name'

**Explanation:** DEC/Test Manager could not read from the specified session file.

**User Action:** Verify that the file exists and check the file's protection and ownership. See the secondary messages for more information.

SESSION\_WRITE, error writing session file 'name'

**Explanation:** DEC/Test Manager could not write data to the specified file.

**User Action:** Verify that the file exists and check the file's protection and ownership. See the secondary messages for more information.

SETMODEERR, could not change terminal characteristics for 'terminal'

**Explanation:** A system service failed to change the specified terminal's device characteristics.

**User Action:** Verify that the terminal exists and check the terminal's protection and ownership.

SETTRMERR, could not change terminal characteristics for 'terminal'

**Explanation:** A system service failed to change your terminal's characteristics.

**User Action:** Verify the terminal's protection and ownership.

SIGSTATUS, routine\_name Signalling on error returned by routine\_call

**Explanation:** This is an internal DEC/Test Manager DECwindows condition indicating that a severe error has occurred.

**User Action:** Submit a Software Performance Report (SPR) specifying this and any additional error messages that are seen.

SIGTRACE, routine\_name is signalling an exit

**Explanation:** This is an internal DEC/Test Manager DECwindows condition.

**User Action:** Submit a Software Performance Report (SPR) specifying this and any additional error messages that are seen.



SIMTERM\_CLS, could not deallocate simulated terminal

**Explanation:** An error occurred while terminating terminal simulation.

**User Action:** See the secondary messages for more information.

SIMTERM\_ERR, error simulating terminal

**Explanation:** An error occurred while simulating your terminal.

**User Action:** See the secondary messages for more information.

SIMTERM\_OPN, could not create simulated terminal

**Explanation:** DEC/Test Manager could not simulate your terminal.

**User Action:** See the secondary messages for more information.

STARTDELHIS, no deletable history records before 'date'

**Explanation:** You specified a date for the /BEFORE qualifier prior to the date of the first history record that can be deleted. The CREATE LIBRARY record cannot be deleted.

**User Action:** Correct the date and enter the command again.

STARTHIS, library history starts at 'date'

**Explanation:** You specified a date for the /BEFORE qualifier prior to the date the library was created.

**User Action:** Correct the date and enter the command again.

STARTIOFAIL, START\_IO FAILED with status = ##

**Explanation:** This is an internal DEC/Test Manager DECwindows condition.

**User Action:** Submit a Software Performance Report (SPR).

STRTOOLONG, string of 'count' characters is too long

**Explanation:** The name you specified is too long.

**User Action:** Enter the command again with a shorter name.

SUBMITTED, collection 'name' submitted

**Explanation:** DEC/Test Manager submitted the specified collection for batch processing.

SUBSYSONLY, EXIT can only be used to leave the subsystem level

**Explanation:** The EXIT command is valid only when you are using DEC/Test Manager as a subsystem or when you are in the Review subsystem.

SUCCEEDED, the comparison for the test 'name' succeeded

**Explanation:** The result file for the specified test matched its benchmark file.

TASKSTRTOOBIG, DECnet Task string is too big

**Explanation:** This is an internal DEC/Test Manager DECwindows condition.

**User Action:** Submit a Software Performance Report (SPR).

TEST\_CONFLICT, DTM test already running

**Explanation:** Cannot record or play a DECwindows test on the default DECwindows server because a DEC/Test Manager DECwindows test is already being recorded or played on the current display.

**User Action:** Wait for the current recording or play operation to terminate.

TESTNOTRUN, test 'name' was not run

**Explanation:** Execution of the collection stopped before all tests had executed.

**User Action:** See Chapter 5 for instructions for reviewing a partially run collection and for reexecuting the tests which did not execute.

TIMEORDER, BEFORE and SINCE time values cannot be resolved

**Explanation:** You specified an incorrect sequence of time values for the /BEFORE and /SINCE qualifiers. (The /SINCE qualifier value must indicate a time prior to that indicated by the /BEFORE qualifier value.)

**User Action:** Verify that you are entering the correct values, then reenter the command.

TOODEEP, eighth-level directory 'directory\_spec' one level too deep

**Explanation:** You specified an eighth-level directory on the CREATE LIBRARY command, which is the deepest directory level RMS allows. You cannot create a DEC/Test Manager library in an eighth-level directory because DEC/Test Manager cannot then create collection subdirectories.

**User Action:** Create another directory no deeper than seven levels.

TOOLONG, 'name' is too long, maximum of 'number' characters

**Explanation:** The specified expression is too long.

**User Action:** Enter the command again with a shorter expression.

TRMCTRL, control characters must be in range ^A through ^Z

**Explanation:** You specified a termination character as an out-of-range control character. Valid control characters include those between CTRL/A and CTRL/Z, inclusive.

**User Action:** Correctly specify the termination character.

TRMRANGE, termination character must be in range 0-255

**Explanation:** You specified a termination character that was either greater than 225 or negative. Valid values include decimal ASCII values between 0 and 255, inclusive.

**User Action:** Correctly specify the termination character.

TRMSYNTAX, specify termination character as single character or ^x or decimal ASCII value

**Explanation:** You specified an invalid termination character. Valid termination characters include single characters, control characters between CTRL/A and CTRL/Z, inclusive, and decimal ASCII values between 0 and 225, inclusive.

**User Action:** Correctly specify the termination character.

TRYAGNLAT, please try again later

**Explanation:** Your library is currently locked by another user.

**User Action:** Wait and enter the command again later.

TTY\_READ, error reading from tty: device

**Explanation:** An error occurred while reading from your terminal.

**User Action:** See the secondary messages for more information.

TTY\_WRITE, error writing to tty: device

**Explanation:** An error occurred while writing to your terminal.

**User Action:** See the secondary messages for more information.

UNDEFLIB, DEC/Test Manager library is now undefined

**Explanation:** You do not have a current DEC/Test Manager library.

**User Action:** Enter a SET LIBRARY command to establish a current DEC/Test Manager library.

UNKLNKSTATE, Network link is in unknown state

**Explanation:** This is an internal DEC/Test Manager DECwindows condition.

**User Action:** Submit a Software Performance Report (SPR) specifying this and any additional error messages that are seen.

UNKNOWN\_SEQ, unknown control or escape sequence 'string'

**Explanation:** Your program output an ASCII control character or an escape sequence that DEC/Test Manager did not understand. DEC/Test Manager could not determine its effect on your terminal screen.

**User Action:** Examine your session file to find the control character or escape sequence DEC/Test Manager did not understand. Verify that your program is outputting the correct escape sequence for the terminal on which it is running. For example, is it sending a VT100 escape sequence to a VT52 terminal? Verify that you are running your program and test on an appropriate terminal that is supported by Digital. Verify that you are not using a terminal feature that this version of DEC/Test Manager does not support. Verify that you are not sending ReGIS or SIXEL codes to your terminal.

UNMATQUOTE, unmatched quote character in expression 'string'

**Explanation:** DEC/Test Manager detected a missing closing quote character ( ' or ").

**User Action:** Enter the command again with a correctly quoted string.

UNSUCCESS, the comparison for the test 'name' was unsuccessful

**Explanation:** The result file for the specified test did not match its benchmark file.

UNSUPFRMT, format of file 'name' not supported by COMPARE

**Explanation:** One of the files DEC/Test Manager is attempting to compare is not a text file.

UNWINDFAILED, SYS\$UNWIND Failed

**Explanation:** This is an internal DEC/Test Manager DECwindows condition.

**User Action:** Submit a Software Performance Report (SPR).

UPDATED, the benchmark for test 'name' has been updated

**Explanation:** DEC/Test Manager replaced the current benchmark file with the result file.

UPDATERR, error in attempt to update the benchmark file for 'result-description-name'

**Explanation:** DEC/Test Manager could not replace the benchmark file with the result file, or DEC/Test Manager could not delete the result file.

**User Action:** See the secondary message for more information.

UPDATES, 'count' updates completed

**Explanation:** DEC/Test Manager updated the specified benchmark files.

UPDNOTNEC, update not necessary for 'result-description-name'

**Explanation:** You tried to update the benchmark file for a successful or updated test.

USERECOVER, use DEC/Test Manager VERIFY/RECOVER

**Explanation:** Your library is in an inconsistent state.

**User Action:** Use the VERIFY/RECOVER command to restore your library to a usable state.

USESETLIB, use DEC/Test Manager SET LIBRARY

**Explanation:** Your DEC/Test Manager library is undefined.

**User Action:** Use the SET LIBRARY command to select a library and reenter the command.

VALREQUIRED, value required for variable 'name' in this context

**Explanation:** When you include the /VARIABLE qualifier with the CREATE COLLECTION command, you must include a variable value for every variable name you list.

**User Action:** Specify a variable value for every variable name you include.

VARCONFLICT, cannot change variable 'name' from 'null string/non-numeric' to null numeric

**Explanation:** You cannot create a numeric symbol variable with a null value.

**User Action:** Examine the use of this variable and determine whether the attempted operation makes sense for this variable.

VARNOVAL, 'name' variable has no default value. Must associate one with this test

**Explanation:** The specified variable does not have a default value.

**User Action:** Use the MODIFY VARIABLE command to associate a default value with the variable.

VERARC, archive list verified

**Explanation:** This phase of the VERIFY command completed successfully.

VERCOL, collection list verified

**Explanation:** This phase of the VERIFY command completed successfully.

VERCOLDIR, collection directory structure verified

**Explanation:** This phase of the VERIFY command completed successfully.

VERFRE, free space list verified

**Explanation:** This phase of the VERIFY command completed successfully.

VERGRO, group list verified

**Explanation:** This phase of the VERIFY command completed successfully.

VERHEAD, user header information verified

**Explanation:** This phase of the VERIFY command completed successfully.

VERIFIED, DEC/Test Manager library 'directory-spec' verified

**Explanation:** Your DEC/Test Manager library has been successfully verified.

VERNOTALL, explicit version numbers not allowed in input file specifications

**Explanation:** You cannot include a version number on a file specification entered as a parameter to a DEC/Test Manager command.

**User Action:** Remove the version number references from all file specifications and enter the command again.

VERSPACE, contiguous space verified

**Explanation:** This phase of the VERIFY command completed successfully.

VERSTR, string list verified

**Explanation:** This phase of the VERIFY command completed successfully.

VERTD, test description list verified

**Explanation:** This phase of the VERIFY command completed successfully.

VERVAR, variables list verified

**Explanation:** This phase of the VERIFY command completed successfully.

WAITFRERR, unable to wait for a system event flag to set

**Explanation:** A system service failed while waiting for an event flag.

**User Action:** Check your quotas and SYSGEN parameters. Consult with your system manager.

WAITING, DEC/Test Manager library 'name' is still in use

**Explanation:** Someone else is still using your library, and your command cannot yet be executed.

**User Action:** DEC/Test Manager will automatically resume execution of your command when the library is free.

WLDNOTALLOWED, wildcard in expression 'name' not allowed

**Explanation:** You cannot include a wildcard character (\* or %) in the specified parameter.

**User Action:** Enter the command again with a valid parameter.

XINFOFAIL, Xtrap\_info request to Xtrap failed

**Explanation:** This is an internal DEC/Test Manager DECwindow condition.

**User Action:** Submit a Software Performance Report (SPR) specifying this and any additional error messages that are seen.

YOUDEL, you must manually delete the collection files

**Explanation:** DEC/Test Manager could not delete all the files in the specified collection.

**User Action:** Use the DCL DELETE command to delete the remaining files.



YOUDELCRE, you must manually delete and re-create this collection

**Explanation:** The original CREATE COLLECTION command was more than 255 characters long. Due to restrictions in the command line interface (CLI), you cannot currently use the RECREATE command for this collection.

**User Action:** Use the DELETE COLLECTION command to manually delete the existing collection. Then reenter the original CREATE COLLECTION command line.

ZLENBLO, a zero length block was found during pass 2

**Explanation:** The library structure contains an error.

**User Action:** Use the VERIFY/RECOVER command to restore your library to a usable state.



**Access Control List (ACL)**

A VMS System protection scheme that grants or denies access to files based on a list of users. With ACLs, you can specify access for a set of users who are not in the same UIC group. For more information on ACL protection, see the *VMS DCL Concepts Manual*. See also **User identification code (UIC)**.

**Benchmark directory**

A VMS directory or VAX DEC/Code Management System (CMS) library that is used to store benchmark files in the current DEC/Test Manager library.

**Benchmark file**

A file that contains the expected results of a correctly completed test. A benchmark file can be created using the UPDATE command during a Review session. It can also be created when an interactive test is recorded in a Session file. You can also create benchmark files manually, but it is not recommended.

**CMS class**

A set of element generations in a VAX DEC/Code Management System (CMS) library with only one generation per element. See **CMS element**.

**CMS element**

A single file stored in a VAX DEC/Code Management System (CMS) library.

**CMS generation**

A representation of a phase in the development of an element in a VAX DEC/Code Management System (CMS) library. Every time you retrieve and then return an element to the library, a new generation is created. Any generation of an element can be retrieved; each generation reflects the changes that were made at that particular point in development.

**CMS library**

The largest group of files that the VAX DEC/Code Management System (CMS) organizes.

**CMS line of descent**

A series of generations of an element, created by successive reservation and replacement transactions within a VAX DEC/Code Management System (CMS) library.

**CMS variant line of descent**

A line of descent separate from the main line of descent—an alternate development path within a VAX DEC/Code Management System (CMS) library. Generation numbers of variant line generations consist of combinations of numbers and variant letters.

**Collection**

One or more tests selected for execution as a set. Tests can be executed only within the context of a collection.

**Collection command file**

A command file created by DEC/Test Manager to execute a collection.

**Collection epilogue file**

A VMS command procedure, created by you using a text editor, that contains the commands that are to be executed after executing the collection command file (see **Collection command file**). You typically use the collection epilogue file to delete directories created specifically for the process or for various other cleanup operations.

**Collection expression**

A command parameter specifying one or more collections.

**Collection name**

A command parameter or name specifying a particular collection.

**Collection prologue file**

A VMS command procedure, created by you using a text editor, that contains the commands that are to be executed prior to executing the collection command file (see **Collection command file**).

**Collection subdirectory**

A subdirectory of the library, created by DEC/Test Manager, where collection-specific files are stored.

**Collection summary**

A summary of the results of a collection run that is displayed when you invoke the SHOW/SUMMARY command in the Review subsystem.

**Command file**

A VMS command procedure, created by you using a text editor, that contains the commands to run the application in a noninteractive environment.

**Comparison status**

The result description status of a test: Comparison Aborted, New Test, Not Run, Successful, Unsuccessful, Updated.

**Difference file**

A file created by DEC/Test Manager from the results of a comparison of the results file to the benchmark file. DEC/Test Manager generates a difference file only if the result file and benchmark file do not match.

**Display device**

A terminal or DECwindows workstation screen.

**DTM\$COLLECTION\_NAME**

A global VMS symbol that is defined by DEC/Test Manager before a collection prologue file executes. This symbol contains the name of the collection and may be used in prologue, epilogue, and command files during the execution of a collection.

**DTM\$INIT**

A logical name assigned by a user to a DEC/Test Manager initialization file.

**DTM\$LIB**

A logical name assigned by DEC/Test Manager to the current DEC/Test Manager library.

## **DTM\$RESULT**

A logical name assigned to the result file for a specific test that is used in prologue, epilogue, and command files during the execution of the test epilogue file.

## **DTM\$TEST\_NAME**

A global VMS symbol that is defined by DEC/Test Manager before a test prologue file executes. This symbol contains the name of the test and may be used in prologue, epilogue, and command files during the execution of a test.

## **Epilogue file**

See **Collection epilogue file** or **Test epilogue file**.

## **Expression**

A command parameter specifying multiple instances of one or more parameters in a single parameter field. An expression can consist of a name, or a list of names separated by commas. You can use wildcards. A result description expression cannot consist of a list separated by commas.

## **Field**

A test description element that DEC/Test Manager uses to associate specific information with a test description. The test description fields are test name, template, benchmark, prologue, epilogue, variables, groups, filters, and remark. See also **Test description** and **Field value**.

## **Field value**

A value for a test description field. When you supply a value for a test description field, you associate specific files, variables, filters, or other attributes with the associated test. See also **Test description** and **Field**.

## **Filter**

A means for substituting constant values for the following run-time variables: directory names, file names, version numbers, date, time, and trace back information.

## **Global variable**

A DCL symbol or logical name that is accessible by any template, prologue file, and epilogue file in a collection. All global variables in the current DEC/Test Manager library are defined at the beginning of every collection run, whether or not the variable is used in the collection.

**Group**

A named set of test descriptions (usually having common characteristics) that can be manipulated as a unit.

**Group expression**

A command parameter specifying one or more groups.

**Group hierarchy**

The relationship among a parent group and all subgroups under it.

**Group name**

A command parameter or name specifying a particular group of test descriptions.

**History**

A date- and time-stamped record of commands that change the DEC/Test Manager library. For example, CREATE and DELETE commands are logged; SHOW commands are not logged. The history is created when you create a DEC/Test Manager library.

**Initialization file**

A DEC/Test Manager command file to be executed whenever DEC/Test Manager is invoked as a subsystem.

**Input file**

A translated session file containing a text representation of all user input, nonprinting control characters, and recording functions in a session file for an interactive terminal test. An input file can be edited.

**Interactive test**

A test whose template is a session file. See also **Session file** and **Noninteractive test**.

**Keypad key**

The editing and numeric keypads, as well as the function keys. These keys are user-definable using the DEFINE/KEY command.

**Keysym**

A value associated with a key on the keyboard. The key values are translated by means of the KEYSYMDEF.H translation table file and enable DEC/Test Manager to use any keyboard.

**Library**

A VMS directory containing the DEC/Test Manager collection subdirectories, result files, difference files, and, optionally, benchmark and session files.

**Local variable**

A DCL symbol or logical name that is defined only while the test with which it is associated is running. Local variables can be used by the prologue, template, and epilogue files associated with this test.

**Mask**

A mask is a user-defined area on a DECwindows benchmark image that is ignored when the results of a test are compared against the benchmark image. Masks are created using the DEC/Test Manager mask editor.

**Mask Editor**

A utility for creating masks on benchmark images.

**Noninteractive test**

A test whose template is a command file. See also **Interactive test**.

**Object expression**

A command parameter specifying one or more tests, groups of tests, collections of tests, or variables. Object expressions encompass all types of DEC/Test Manager entities.

**Output files**

One or more files associated with each test after a collection has been executed and compared; output files can be benchmark, result, or difference files.

**Panning**

Moving a DECwindows workstation screen image with the pointer; the image remains in the same position relative to the pointer.

**Parameter qualifier**

A qualifier (either /GROUP or /TEST\_DESCRIPTION) used after each item in a test group expression to identify the item as either a test name or a group name.



**Primary reviewer**

A person who enters the REVIEW command without the /READ\_ONLY qualifier. Only one person at a time can be the primary reviewer of a collection. This reviewer can use all Review subsystem commands. See also **Read-only reviewer**.

**Prologue file**

See **collection prologue file** or **test prologue file**.

**Read-only reviewer**

A person who enters the REVIEW command with the /READ\_ONLY qualifier. A read-only reviewer can peruse the result descriptions and print files, but cannot make any changes to the result descriptions. A read-only reviewer cannot issue the UPDATE or INSERT commands.

**Record type**

A 1-byte indicator that describes the contents of a session file record. See also **Terminal characteristics block**.

**Regression testing**

A testing method that ensures that software being developed or modified runs correctly. As new features are added, the software is repeatedly tested to verify that new features do not affect the correct execution of the previously tested features. When errors exist, the software is said to have regressed.

**Remark**

A comment, associated with a DEC/Test Manager command, that is recorded in the DEC/Test Manager history file.

**Result description**

A summary of test results accessible from the Review subsystem. DEC/Test Manager generates a result description for every test description in a collection.

**Result description name**

A command parameter or name specifying a particular result description. The result description name for a test is the same as its test name.

**Result description expression**

A command parameter specifying one or more result descriptions. A result description expression cannot consist of a list separated by commas. It can consist of a result description name, a wildcard character, or a wildcard character used in conjunction with a full or partial result description name.

**Result file**

A file containing the results of a test's execution and accessible only from within the Review subsystem.

**Review subsystem**

The DEC/Test Manager subsystem in the terminal environment that enables you to examine and manipulate the results of running a collection of tests. The Review subsystem is not used for reviewing DECwindows tests; they are reviewed using the DECwindows DEC/Test Manager user interface.

**Scrolling**

Moving a DECwindows workstation screen image with sliders or increment arrows in the scroll bars.

**Session file**

A file containing a recording of an interactive terminal or DECwindows session. It contains all actions that you enter until you end the session.

**Special strings**

Text representations in input files for the nonprinting characters and recording functions; they replace all nonprinting characters and recording functions found in session files.

**Subgroup**

A group that is part of another group.

**Template directory**

A VMS directory or VAX DEC/Code Management System (CMS) library used for template files in the current DEC/Test Manager library.

**Template file**

A VMS command procedure that executes a noninteractive test or a session file.

**Terminal characteristics block**

The first 12-byte record in a session file; it describes the type of terminal on which the terminal session was recorded and the characteristics of that terminal. The terminal characteristics block is described in the *VMS I/O User's Reference Manual: Part 1*.

**Termination character**

The character which, when entered twice, terminates the recording of an interactive terminal session. The default termination character is CTRL/P.

**Test**

A command procedure or recorded session file that executes applications for the purpose of regression testing.

**Test description**

A collection of fields for which you supply values that point to the files and other entities associated with the test. A test description contains all the information DEC/Test Manager needs to run a particular test. Each test must have a corresponding test description. See also **Field** and **Field value**.

**Test epilogue file**

An optional command file you specify, such as a filter file, that is associated with a test and runs after the test executes. You typically use a test epilogue file to modify the results file of run-dependent data.

**Test expression**

A command parameter specifying one or more test descriptions.

**Test group expression**

A command parameter specifying one or more test descriptions or groups. The parameter qualifiers /GROUP and /TEST\_DESCRIPTION identify the individual items in a test group expression as specifying either groups or test descriptions.

**Test name**

A command parameter or name specifying a particular test description.

**Test prologue file**

An optional command file, such as a setup file, that is associated with a test and runs before the specified test. You typically use the test prologue file to establish any special environment that the test requires such as setting up constants, defining versions, defining logical names, or creating directories. Output from the test prologue file does not appear in the test results.

**Type-ahead**

A function that enables you to enter keystrokes faster than the characters can be sent to the output device. The type-ahead function places the characters generated by the keystrokes into a buffer until system resources allow them to be sent to the output device.

**User identification code**

A code that determines a user's access rights to a file. For more information on UIC protection, see the *VMS DCL Concepts Manual*. See also **Access Control List**.

**Variable**

A DCL symbol or logical name that you define in a DEC/Test Manager library and associate with collections or tests (or both) in that library.

**Variable expression**

A command parameter specifying one or more variables.

**Variable name**

A command parameter or name specifying a particular variable.

**VAX DEC/Code Management System (CMS)**

A software library system that stores files, records changes made to the files, and records user access of the files.

**VAX Performance and Coverage Analyzer (PCA)**

A software development tool that collects performance and coverage data on a program and enables you to interactively analyze that data. When DEC/Test Manager is used with the VAX Performance and Coverage Analyzer, DEC/Test Manager invokes the PCA Collector to gather performance and coverage data while tests are running.

**Wildcard characters**

Characters used to specify one or more command parameter specifications. Use the asterisk (\*) for partial- or full-field substitutions use the percent sign (%) for single-character substitutions.

@file-specification command, CD-7 to CD-8,  
CD-172

## A

---

Accepting unsolicited input, 3-25

Access control entry

See ACE

Access control list

See ACL

ACE (access control entry), 7-7

ACL (access control list), 7-5

using on libraries, 7-7 to 7-8

using on library files, 7-8 to 7-11

Analyzer Filter

See PCA, Analyzer filter

ATTACH, 2-4

ATTACH command, 6-24, CD-9 to CD-10, CD-174  
to CD-175

Attaching to a process, CD-9, CD-174

Automatic comparison abortion, 4-9

Automatic screen compare

See Comparison, automatic

## B

---

BACK command, CD-176

Benchmark directory

See also benchmark file

canceling the default, 7-3, CD-124

default, 7-2

displaying the default, CD-135

establishing the default, CD-123

overriding the default, 7-3, CD-26, CD-87,  
CD-124

setting, 3-3

Benchmark file, 3-17, 7-2

See also Benchmark directory

associating a file specification with, CD-87

creating, CD-211

creating for a new test

in the Review subsystem, 5-22 to 5-24

creating masks, 2-15 to 2-16

default file name, CD-38

displaying, 5-18 to 5-19

for interactive terminal tests, CD-64

the Review subsystem, CD-202

removing a file specification from, CD-87

replacing, CD-87, CD-211

restriction on storing, 3-27

saving multiple versions, CD-124

setting benchmark directories, 7-2 to 7-3

storing

in CMS libraries, 7-3 to 7-4

outside the library, 7-2

updating

in DECwindows, 2-14

in the Review subsystem, 5-21 to 5-22

Benchmark image file, 2-12

Benchmark images

viewing, 2-3

Binary session file, 9-1

## C

---

Callable interface, 10-1 to 10-8

see also DTM\$DTM

error conditions, 10-7

Callback routine

writing, 10-6

Canceling commands, 1-5

## Canceling commands (Cont.)

in the Review subsystem, 5-8

## Checking a library for errors

See Library, verifying

## Cleaning an environment

See Test epilogue file

## CMS, Glossary-1

### commands

CMS CREATE ELEMENT, CD-211

CMS FETCH, CD-197

CMS INSERT/SUPERSEDE, CD-211

CMS REPLACE, CD-211

CMS RESERVE, CD-211

### libraries

storing files in, 7-3 to 7-4

variant line of descent, CD-211

## Code Management System

See CMS

## Collection, 1-4

command file, Glossary-2

comparing partially run, CD-12

creating, 4-1 to 4-3, CD-25

in DECwindows, 2-10

deleting, 4-7 to 4-8, CD-51

displaying summary information, 4-7, CD-136

epilogue file, 4-3, 6-4 to 6-6

sample, 6-5 to 6-6

errors in summary information for read-only

reviewers, 5-7

executing, 4-3 to 4-5

in batch, 4-5, CD-25, CD-120, CD-167

in DECwindows, 2-11

interactively, 4-5 to 4-6, CD-25, CD-119

expression, Glossary-2

incorporating changed files, 4-8

incorporating changes to library, 4-8

name, Glossary-2

organizing tests into, 4-1 to 4-3

prologue file, 4-3, 6-4 to 6-6

sample, 6-5

recomparing partially compared, 4-10 to 5-1

re-creating, 4-8, CD-105

requirements for creation, CD-26

rerunning, CD-167

reviewing, CD-117

partially run, 5-25 to 5-26

stopping execution of, 4-6, 5-25, CD-120,

CD-164

subdirectory, Glossary-3

summary, Glossary-3

types of tests in, 4-5

Collections, 3-1

Collection view, 2-3

Command Correlation, 2-4

Command file, 6-22 to 6-23

creating, 6-22 to 6-23

executing, CD-7, CD-172

Command keysym key, 3-17

Command Keysym key

redefining, 3-20

Command qualifier

using, CD-5

## Commands

See also Review subsystem, commands

ATTACH, CD-9 to CD-10

canceling, 1-5 to 1-6

in the Review subsystem, 5-8

COMPARE, CD-11 to CD-16

control key sequences, 3-17 to 3-19

CONVERT LIBRARY, CD-17 to CD-18

COPY TEST\_DESCRIPTION, CD-19 to CD-23

CREATE COLLECTION, CD-24 to CD-31

CREATE GROUP, CD-32 to CD-33

CREATE LIBRARY, CD-34 to CD-35

CREATE TEST\_DESCRIPTION, CD-36 to CD-42

CREATE VARIABLE, CD-43 to CD-46

DECwindows correlation, 2-4

DEFINE/KEY, CD-47 to CD-50

defining keys for, 6-20 to 6-21

DELETE COLLECTION, CD-51 to CD-52

DELETE GROUP, CD-53 to CD-55

DELETE HISTORY, CD-56 to CD-58

DELETE TEST\_DESCRIPTION, CD-59 to CD-61

DELETE VARIABLE, CD-62 to CD-63

DISPLAY, CD-64 to CD-65

DTM, CD-66 to CD-67

entering, 1-4 to 1-5

EXIT, CD-68

EXTRACT, 9-1, CD-69 to CD-71

@file-specification, CD-7 to CD-8

FILTER, CD-72 to CD-74

general form of, CD-3

HELP, CD-75 to CD-76

INSERT GROUP, CD-77 to CD-79

INSERT TEST\_DESCRIPTION, CD-80 to CD-82

MODIFY GROUP, CD-83 to CD-84

MODIFY TEST\_DESCRIPTION, CD-85 to CD-92

MODIFY VARIABLE, CD-93 to CD-95

parameters, CD-3 to CD-4

PLAY, CD-96 to CD-98

privileges required to use, 7-8 to 7-10 table

RECORD, CD-99 to CD-103

## Commands (Cont.)

- recording key sequences, 3-17 to 3-19
- RECREATE, CD-104 to CD-106
- REMARK, CD-107 to CD-108
- REMOVE GROUP, CD-109 to CD-110
- REMOVE TEST\_DESCRIPTION, CD-111 to CD-113
- RESTORE, 9-4, CD-114 to CD-115
- REVIEW, CD-116 to CD-118
- RUN, CD-119 to CD-122
- SET BENCHMARK\_DIRECTORY, CD-123 to CD-124
- SET EPILOGUE, CD-125 to CD-126
- SET LIBRARY, CD-127 to CD-128
- SET NOBENCHMARK\_DIRECTORY, CD-124
- SET NOEPILOGUE, CD-125
- SET NOPROLOGUE, CD-129
- SET NOTEMPLATE\_DIRECTORY, CD-132
- SET PROLOGUE, CD-129 to CD-130
- SET TEMPLATE\_DIRECTORY, CD-131 to CD-132
- SHOW ALL, CD-133 to CD-134
- SHOW BENCHMARK\_DIRECTORY, CD-135
- SHOW COLLECTION, CD-136 to CD-139
- SHOW EPILOGUE, CD-140
- SHOW GROUP, CD-141 to CD-143
- SHOW HISTORY, CD-144 to CD-147
- SHOW LIBRARY, CD-148
- SHOW PROLOGUE, CD-149
- SHOW TEMPLATE\_DIRECTORY, CD-150
- SHOW TEST\_DESCRIPTION, CD-151 to CD-155
- SHOW VARIABLE, CD-156 to CD-158
- SHOW VERSION, CD-159
- SPAWN, CD-160 to CD-163
- STOP, CD-164 to CD-165
- SUBMIT, CD-166 to CD-168
  - syntax, CD-3 to CD-6
- VERIFY, CD-169 to CD-171

Comment record, 8-5

Compare

- marking a screen, 2-9

COMPARE command, CD-11 to CD-16

Comparing

- interactive tests, CD-12
- test results, 4-9 to 4-10

Comparison

- automatic, 4-9, CD-12
- ignoring special characters during, 4-10
- manual, CD-12
- partially run collections, CD-12

## Comparison (Cont.)

type

- character-by-character, 4-9
- record-by-record, 4-9
- screen-by-screen, 4-9

Comparison aborted comparison status, 4-9 table

Comparison status, 5-3

- specifying result descriptions by, 5-4
- values, 4-9 table

Comparison status qualifiers, 5-6

COMPILE command (LSE), 3-24

Compose Character key, 2-9

Context-sensitive help, 2-2

Control key sequence commands, 3-17 to 3-19

CONVERT LIBRARY command, CD-17 to CD-18

Copying test descriptions, 3-10 to 3-11

- in a terminal environment, 3-10 to 3-11
- restrictions, 3-11

COPY TEST\_DESCRIPTION command, 3-10, CD-19 to CD-23

Create a DECwindows test, 2-5

CREATE COLLECTION command, 3-22, CD-24 to CD-31

CREATE command (DCL), 3-2

CREATE GROUP command, CD-32 to CD-33

CREATE LIBRARY command, 3-3, CD-34 to CD-35

CREATE TEST\_DESCRIPTION command, 3-15, CD-36 to CD-42

CREATE VARIABLE command, CD-43 to CD-46

Creating

- a library, 3-2 to 3-3
- interactive test, 3-16 to 3-19
- noninteractive test, 3-14 to 3-15
- test description, 3-8 to 3-9
- test descriptions
  - in DECwindows, 2-5 to 2-6
- variables, 6-11

Creating informational messages in input files, 9-9 to 9-11

CTRL/P

- See Termination character, default

## D

---

Data record, 9-6

DCL command procedure

- See noninteractive test

DCL prompt, 1-5, 3-3

DEC/Test Manager, 1-1

- DECwindows interface

- See DECwindows interface

## DEC/Test Manager (Cont.)

- displaying version, CD-159
- exiting, CD-68
- invoking, CD-66
- library, 1-3
  - See Library
- messages
  - See Messages
- supported environments, 1-3
- test
  - See Test
- DEC/Test Manager Help, 2-2
- DEC/Test Manager library, 2-4
- DECwindows
  - input file
    - creating, 9-1, 9-2 to 9-4
    - editing, 9-5
    - session file, 9-1
      - creating from an input file, 9-4
  - DECwindows interface, 2-1 to 2-3
    - command correlation, 2-4
    - displaying information, 2-2 to 2-3
    - displaying test results, 2-12 to 2-13
    - initial view, 2-1
    - views, 2-2
  - DECwindows terminal emulator window, 2-1
  - DECwindows test, 3-7
  - DEFINE/KEY, CD-47 to CD-50
    - key names, CD-47 List, CD-178-List
  - DEFINE/KEY command, CD-178
  - DELETE COLLECTION command, CD-51 to CD-52
  - DELETE command (DCL), 5-25
  - DELETE GROUP command, CD-53 to CD-55
  - DELETE HISTORY command, 3-6, CD-56 to CD-58
  - DELETE TEST\_DESCRIPTION command, CD-59 to CD-61
  - DELETE VARIABLE command, CD-62 to CD-63
- Deleting
  - collections, 4-7 to 4-8, CD-51
  - groups, 6-10 to 6-11, CD-53
  - history information, CD-56
  - test descriptions, 3-13
  - test\_descriptions, CD-59
  - variables, 6-13, CD-62
- Device type, 3-25
- Diagnostic messages
  - See Messages
- Difference file, 5-2
  - displaying, 5-19, CD-203
  - restriction on changing highlighting, 5-16

## Differences

- viewing, 2-3
- DIRECTORY command (DCL), 3-24
- DISPLAY/BENCHMARK
  - keypad, 5-12 to 5-14
- DISPLAY command, CD-64 to CD-65
- Display device, 3-16
- Displaying
  - benchmark files, 5-18 to 5-19
  - collection summary, 4-7
  - difference files, 5-19
  - group structure, 6-9 to 6-10
  - history information, CD-144
  - information in DECwindows, 2-2 to 2-3
  - library information, 3-4 to 3-5
  - result files, 5-18 to 5-19
  - test descriptions, 3-10
  - test results, 5-16 to 5-19
    - for interactive tests, 5-18
    - for noninteractive tests, 5-17 to 5-18
  - in DECwindows, 2-12 to 2-13
  - using comparison status, 5-17
  - using output qualifiers, 5-17
- DTM\$COLLECTION\_NAME, 4-4
- DTM\$DTM
  - calling sequence, 10-1
  - command line, 10-2
  - confirmation routine, 10-3 to 10-5
  - initialization flag, 10-6
  - message routine, 10-2
  - output routine, 10-5
  - output width, 10-5
  - prompt routine, 10-2 to 10-3
- DTM\$RESULT, 4-4
- DTM\$TEST\_NAME, 4-4
- DTMSHR.EXE shareable image, 10-8

## E

- 
- Endloop command, 9-5
  - EndLoop command, 9-7
  - Entering commands, 1-4 to 1-5
  - Environment initialization, 3-27
  - Epilogue file
    - adding to a test description, CD-88
    - canceling the default collection, CD-125
    - collection, 6-4 to 6-6
    - displaying the default collection, CD-140
    - establishing the default collection, CD-125
    - overriding the default collection, CD-125
    - removing from a test description, CD-88



## Epilogue file (Cont.)

- replacing for a test description, CD-88
  - storing
    - in CMS libraries, 7-3 to 7-4
    - outside the library, 7-2
  - test, 6-2 to 6-4
  - types of, 6-2 table
  - using, 6-1 to 6-6
- Examining test results, 5-6 to 5-19
- Executing a collection
  - in DECwindows, 2-11
- Executing collections, 4-3 to 4-5
  - sequence of events, 4-4 to 4-5
- EXIT command, CD-68, CD-182
- Exiting
  - from a recording session, 3-19
  - from the Review subsystem, 5-7, CD-182
- Expanding a view, 2-3
- Expression, Glossary-4
- EXTRACT command, 3-22, CD-69 to CD-71
  - creating a DECwindows input file, 9-1
- Extracting an input file, 3-22, 8-13

## F

---

- File specification, CD-6
- Filter, 6-18 to 6-20
  - adding to a test description, CD-39, CD-88
  - Analyzer
    - See PCA, Analyzer filter
  - applying to files, 6-20
  - associating and disabling, 6-19 to 6-20
  - displaying test description, CD-153
  - removing from a test description, CD-90
  - replacing in a test description, CD-88
  - specifying in a terminal environment, 6-18 to 6-19
- FILTER command, CD-72 to CD-74
- FIRST command, CD-184

## G

---

- Global variable, Glossary-4
- Groups, 6-6 to 6-11
  - changing the hierarchy of, CD-110
  - commands for, 6-7 to 6-7 table
  - creating, CD-33
  - creating a hierarchy, 6-8 to 6-9, CD-78
  - deleting, 6-10 to 6-11, CD-54
  - displaying
    - contents, CD-54
    - information about, CD-141

## Groups

- displaying (Cont.)
  - members, CD-54
  - structure, 6-9 to 6-10
- expression, Glossary-5
- hierarchy, Glossary-5
- inserting groups into a group, CD-33, CD-78
- inserting test descriptions into a group, CD-33, CD-81
- modifying, CD-84
- organizing tests into, 6-7 to 6-8
- removing
  - subgroups from a group, CD-110
  - test descriptions from a group, CD-112
  - tests and subgroups, 6-10
- replacing the remark, CD-84
- restrictions when creating, CD-77

## H

---

### Help

- context-sensitive, 2-2
  - online, 1-5, 2-2
- HELP command, CD-75 to CD-76, CD-185
- History, 3-5 to 3-7
  - adding a remark, 3-6, CD-107
  - commenting unusual occurrence in, CD-107
  - deleting history information, 3-6 to 3-7, CD-56
  - displaying summary information, CD-144
  - logging commands in, 3-5

## I

---

- Ignoring masks, 2-16
- Ignoring special characters during comparison, 4-10
- Informational messages
  - in input files, 9-9
- Initialization file, Glossary-5
- Initializing the testing environment, 3-21
- Input file
  - character translations, CD-70
  - creating, 3-22
  - DECwindows
    - changing input event times, 9-11
    - comments, 9-5 to 9-6
    - creating, 9-1
    - creating informational messages, 9-9 to 9-11
    - data record, 9-6
    - editing, 9-5
    - Loop command, 9-7
    - repeating tasks, 9-7 to 9-8

## Input file

### DECwindows (Cont.)

- synchronization record, 9-6, 9-7
- synchronize play back, 9-6 to 9-7

extracting from a session file, CD-70

informational messages, 9-9

recording a session file

- using the INSERT recording function, 8-19

restoring a session file, CD-115

SendConsole command, 9-9

special strings, CD-70

terminal, 8-11 to 8-21

- contents of, 8-1, 8-11

creating, 8-11

creating with a text editor, 8-17

definition of, 8-1

entering comments in, 8-15

extracting from a session file, 8-13, 8-16

in a CMS library, 8-16

nesting, 8-19

recording a session file, 8-17

special strings, 8-12

terminal characteristics, 8-19

using a nondefault termination character,  
8-16

termination character, CD-70

INSERT command, CD-187

INSERT GROUP command, CD-77 to CD-79

INSERT TEST\_DESCRIPTION command, CD-80 to  
CD-82

Interactive test, 3-7

creating, 3-16 to 3-19

DECwindows

replaying, 3-23

displaying test results, 5-18

processing considerations, 3-23 to 3-25

replaying, 3-22 to 3-23

supported screen sizes, 3-20

terminal

replaying, 3-23

Invoking

DEC/Test Manager subsystem, CD-66

Review subsystem, CD-117

## K

---

### Keypad

defining, 6-20 to 6-21

command keys, 6-21

GOLD command keys, 6-21

Display benchmark, 5-12 to 5-14

### Keypad (Cont.)

Review subsystem, 5-10 to 5-12

displaying defaults, CD-179

saving key definitions, 6-21

Show benchmark, 5-12 to 5-14

Show differences, 5-14 to 5-16

Show result, 5-12 to 5-14

### Keys

defining, CD-47

display default key definitions, CD-48, CD-179

saving key definitions, CD-48, CD-179

Keysym key, 3-17

redefining, 3-20

## L

---

LAST command, CD-190

Library, 1-3, 3-1, 3-1 to 3-3, 7-6

consolidating structure, 7-2

converting

See CONVERT LIBRARY command

correcting errors in, CD-170

correcting invalid, 7-1 to 7-2

creating, 3-2 to 3-3, CD-35

displaying information, 3-4 to 3-5

displaying summary information, CD-133

evaluating structure, 7-1, CD-169

file storage outside, 7-2

overview, 3-1

privileges required to use, 7-6

table

recovering, CD-169

selecting, 3-3

setting, 3-3, CD-127

setting without verification, CD-128

specifying a UIC protection mask, 7-5

storing

files in CMS, 7-3 to 7-4

using ACLs, 7-7 to 7-11

verifying, CD-169

Library specification, 3-2

LINK command (DCL), 10-8

Linking with the DEC/Test Manager image, 10-8

Local variable, Glossary-6

Locating test results

in a terminal environment, 5-8 to 5-9

in DECwindows, 5-9

Loop command, 9-5, 9-7

Loops

DECwindows input file, 9-7

## M

---

- MAIL\_COLLECTION collection, 2-3
- Manual screen compare
  - See Comparison, manual
- Marks screen for comparison, 2-9
- Mask Editor, 2-15
  - leaving, 2-16
- Masks, 2-15 to 2-16
  - deleting, 2-16
  - ignoring, 2-16
  - moving, 2-16
  - saving with a benchmark image, 2-16
- Menu bar, 2-1
- Message handler
  - LIB\$ESTABLISH routine, 10-8
  - mechanism array, 10-8
  - routine, 10-7 to 10-8
  - signal array, 10-8
- Messages, A-1 to A-65
  - display of, A-1
  - fields in, A-1
  - format of, A-1
  - name, A-1
  - severity code, A-1, A-2
  - text, A-1
- MODIFY GROUP command, CD-83 to CD-84
- Modifying
  - test descriptions, 3-12 to 3-13
  - variable characteristics, 6-13
  - variables, 6-11
- MODIFY TEST\_DESCRIPTION command, CD-85 to CD-92
- MODIFY VARIABLE command, CD-93 to CD-95

## N

---

- Nesting loop sequences, 9-5, 9-7
- New test comparison status, 4-10 table
- NEXT command, CD-191
- Noninteractive test, 3-7
  - creating, 3-14 to 3-15
  - displaying test results, 5-17 to 5-18
  - test description, 3-15
  - writing, 3-14
  - writing a template file, 3-14
- Not run comparison status, 4-10 table

## O

---

- Object expression, Glossary-6

- Online help
  - See Help
- Organizing tests into collections, 4-1
- Output files, 5-3
  - qualifiers, 5-6
  - types of, 5-3 table

## P

---

- Panning, 2-12
- Parameter, CD-3
- Parameter qualifier, Glossary-6
  - using, CD-5
- PCA, Glossary-10
  - Analyzer filter, CD-194
  - invoking, CD-194
- PCA command, CD-194
- Playback system, 9-6
- PLAY command, 3-22, CD-96 to CD-98
- Playing DECwindows Tests
  - restriction, 3-25
- Primary reviewer, 5-7, CD-118
- PRINT command, CD-195
- Printing test results, 5-20
- Process
  - attaching to a, CD-9
- Prologue file
  - adding to a test description, CD-90
  - canceling in a test description, CD-90
  - canceling the default collection, CD-129
  - collection, 6-4 to 6-6
  - displaying the default collection, CD-149
  - establishing the default collection, CD-129
  - overriding the default collection, CD-129
  - removing from a test description, CD-90
  - replacing for a test description, CD-90
  - storing
    - in CMS libraries, 7-3 to 7-4
    - outside the library, 7-2
  - test, 6-2 to 6-4
  - types of, 6-2 table
  - using, 6-1 to 6-6

## Q

---

- Qualifier
  - placement of, CD-3
  - using, CD-4 to CD-5
    - command qualifiers, CD-5
    - parameter qualifiers, CD-5

## R

- Read-only reviewer, 5–7, CD–118
- Real-time, 3–26
- Recomparing partially compared collections, 4–10 to 5–1
- RECORD command, 9–4, CD–99 to CD–103
- Record dialog box, 2–6
- Recording
  - DECwindows tests, 2–6 to 2–9, 3–21 to 3–22
  - exiting a session, 3–19
  - restrictions
    - accepting unsolicited input, 3–25
    - CTRL/C, 3–24
    - CTRL/Y, 3–24
    - device type, 3–25
    - terminal characteristics, 3–25
    - timing-dependent applications, 3–24
    - type-ahead, 3–24
  - restrictions on interactive terminal sessions, 3–23
    - unsolicited input, 3–25
    - using type-ahead, 3–24
  - terminal tests, 3–20 to 3–21
  - tests, 3–17 to 3–19
- Recording key sequence commands, 3–17 to 3–19
- Record type, Glossary–7
- Recovering a library with errors
  - See Library, recovering
- Recovering invalid libraries, 7–1 to 7–2
- RECREATE command, CD–104 to CD–106
- Re-creating a collection, 4–8
- Regression testing
  - typical steps, 1–1 to 1–2
- Remark, 1–6
  - syntax, CD–5
- REMARK command, 3–6, CD–107 to CD–108
- REMOVE GROUP command, CD–109 to CD–110
- REMOVE TEST\_DESCRIPTION command, 3–13, CD–111 to CD–113
- Removing tests or subgroups from groups, 6–10
- Replaying interactive test, 3–22 to 3–23
  - DECwindows, 3–23
  - terminal, 3–23
- RESTORE command, 9–4, CD–114 to CD–115
- Result description, 5–2, CD–117
  - specifying, 5–4 to 5–5
    - by comparison status qualifiers, 5–4
    - by output file qualifiers, 5–4
    - by result description expression, 5–4
- Result file, 5–3 table
  - displaying, 5–18 to 5–19, CD–204
- Result file (Cont.)
  - restriction on storing, 3–27
- Result image file, 2–12
- Result images
  - viewing, 2–3
- REVIEW command, CD–116 to CD–118
- Reviewing
  - concepts, 5–1 to 5–3
  - DECwindows test results, 2–12 to 2–13
  - examining test results, 5–6 to 5–19
  - partially run collections, 5–25 to 5–26
  - primary reviewer, 5–7
  - read-only reviewer, 5–7
  - test results, 5–1, CD–117
- Review subsystem
  - canceling commands, 5–8
  - commands
    - ATTACH, CD–174 to CD–175
    - BACK, CD–176 to CD–177
    - DEFINE/KEY, CD–178 to CD–181
    - EXIT, CD–182
    - FIRST, CD–184
    - HELP, CD–185 to CD–186
    - INSERT, CD–187 to CD–188
    - LAST, CD–190
    - NEXT, CD–191 to CD–193
    - PCA, CD–194
    - PRINT, CD–195 to CD–197
    - SELECT, CD–199
    - SHOW, CD–200 to CD–202
    - SPAWN, CD–206 to CD–209
    - UPDATE, CD–210 to CD–212
  - comparison status qualifiers on the PRINT command, CD–196
  - creating
    - group on exit, CD–188
    - new benchmark file, CD–211
  - displaying
    - benchmark file, CD–202
    - collection summary information, CD–204
    - difference file, CD–203
    - information about the collection, CD–201
    - output files for a collection, CD–201
    - result file, CD–204
    - test results, 5–16 to 5–19
  - exiting, 5–7, CD–182
  - grouping tests, CD–188
  - invoking, 5–6, CD–117
    - as a read-only reviewer, CD–117
    - as the primary reviewer, CD–117
    - PCA, CD–194

## Review subsystem (Cont.)

- keypad, 5-10 to 5-16
  - defining keys, CD-179
  - displaying defaults, CD-179
  - help, CD-179
  - replacing defaults, CD-179
- locating
  - results, 5-8 to 5-9
- locating a specified result description, CD-199
- output file qualifiers on the PRINT command, CD-196
- printing
  - files, CD-196
  - test results, 5-20
- reviewing
  - a collection, CD-117
  - performance and coverage data, CD-194
- reviewing performance and coverage data
  - See also, PCA
- spawning a subprocess, CD-207
- updating a benchmark file, CD-211
- RUN command, 3-22, CD-119 to CD-122
  - executed from batch, CD-120
- Running collections, 4-5 to 4-6

## S

---

### Sample

- Adding loops to a DECwindows input file, 9-8
- creating informational messages in a DECwindows input file, 9-9 to 9-11

### Sample DECwindows session, 2-4

### Samples

- collection epilogue file, 6-5 to 6-6
- collection prologue file, 6-5
- copying test descriptions, 3-11
- create collection, 4-2
- creating
  - benchmark file, 5-23 to 5-24
  - groups, 6-7 to 6-8
  - input file with an editor, 8-17
  - library, 3-2
  - noninteractive test description, 3-15
- DECwindows input file, 9-2 to 9-4
- deleting
  - collection, 4-8
  - groups, 6-10 to 6-11
  - history information, 3-6
  - test description, 3-13
- difference file screen 0, 5-19 figure

## Samples

### displaying

- collection summary, 4-7
- group structure, 6-9 to 6-10
- history, 3-5 to 3-6
- library summary information, 3-5
- test descriptions, 3-10

DTM\$COLLECTION\_NAME, using, 6-6

DTM\$RESULT, using, 6-16 to 6-17

DTM\$TEST\_NAME, using, 6-15 to 6-16

extracting an input file, 8-11

initialization command file, 6-23

interactive terminal session, 1-6 to 1-10

invoking the Review subsystem, 5-6 to 5-7

locating result descriptions with comparison status, 5-9

modifying a test description, 3-12, 3-13

noninteractive test file, 3-14

noninteractive test template file, 3-15

overriding variables, 6-14

redefining the termination character, 3-19 to 3-20

result description, 5-2

running a collection, 4-6

setting

- benchmark directory, 3-3

- library, 3-3

- template directory, 3-4

stopping collection execution, 4-6

terminal session, 1-6 to 1-10

test epilogue file, 6-3 to 6-4

test prologue file, 6-3

updating a benchmark file, 5-21 to 5-22

### Scrolling, 2-12

### Security features, 7-5 to 7-11

- ACL protection, 7-7 to 7-11

- libraries, 7-7 to 7-8

- library files, 7-8 to 7-11

- UIC protection, 7-5 to 7-7

### SELECT command, CD-199

### Selecting a view, 2-3

### SendConsole command, 9-5

- DECwindows input file, 9-9

### Session file, 3-7, 3-16, CD-70, CD-115

- creating input files from, 3-22

- DECwindows, 9-1 to 9-11

- creating from an input file, 9-4

- terminal, 8-1 to 8-35

- comment record, 8-5

- contents of, 8-1

- effects of

- asynchronous events on, 8-6

## Session file

### terminal

#### effects of (Cont.)

- program output on, 8-6
- system response on, 8-5
- terminal driver on, 8-5, 8-6
- typing speed on, 8-5

#### extracting an input file, 8-16

#### first record, 8-7

#### in a CMS library, 8-16

#### modifying, 8-9

- BEGIN\_COMPARE records, 8-10
- COMPARE\_SCREEN records, 8-10
- END\_COMPARE records, 8-10
- INPUT records, 8-10
- OUTPUT records, 8-10
- TIMING records, 8-11
- WAIT records, 8-11

#### record, 8-8

#### structure of, 8-6

#### type, 8-8 table

#### recording from an input file, 8-17

- using the INSERT recording function, 8-19

#### restrictions on changing timing, 8-11

#### special strings, 8-1

#### structure, 8-5

#### terminal characteristics, 8-19

#### terminal characteristics block, 8-2, 8-7

#### translations performed when extracted, 8-13

### SET BENCHMARK\_DIRECTORY command, CD-123 to CD-124

### SET EPILOGUE command, CD-125 to CD-126

### SET FILE command (DCL), 7-7, 7-8

### SET LIBRARY command, 3-3, CD-127 to CD-128

### SET PROLOGUE command, CD-129 to CD-130

### SET PROTECTION command (DCL), 7-5

### SET TEMPLATE\_DIRECTORY command, 3-4, CD-131 to CD-132

### Setting

#### benchmark directories, 3-3

#### libraries, 3-3

#### template directories, 3-4

### Setting up an environment

#### See Test prologue file

### SHOW ALL command, CD-133 to CD-134

### SHOW/BENCHMARK

#### command, 5-10

#### keypad, 5-12 to 5-14

### SHOW BENCHMARK\_DIRECTORY command, CD-135

### SHOW COLLECTION command, CD-136 to CD-137

### SHOW command, CD-200

### SHOW/DIFFERENCES

#### command, 5-10

#### keypad, 5-14 to 5-16

### SHOW EPILOGUE command, CD-140

### SHOW GROUP command, CD-141 to CD-143

### SHOW HISTORY command, 3-5, CD-144 to CD-147

### Showing a collection summary

#### See Collections, displaying a summary

### SHOW LIBRARY command, 3-4, CD-148

### SHOW PROLOGUE command, CD-149

### SHOW/RESULT

#### command, 5-10

#### keypad, 5-12 to 5-14

### SHOW TEMPLATE\_DIRECTORY command, CD-150

### SHOW TEST\_DESCRIPTION command, CD-151 to CD-155

### SHOW VARIABLE command, CD-156 to CD-158

### SHOW VERSION command, CD-159

### Software Performance Report

#### displaying version, CD-159

### SPAWN, 2-4

### SPAWN command, 6-24, CD-160 to CD-163, CD-206

### Special strings, Glossary-8

#### arrow key codes, 8-14

#### editing key codes, 8-14

#### format, 8-12

#### 8-bit control characters, 8-13

#### control characters, 8-13

#### nonprinting characters, 8-13

#### formats for nonprinting characters, 8-13

#### function key codes, 8-14

#### key names, 8-14

#### keypad key codes, 8-14

#### recording functions, 8-14

#### types recognized by DEC/Test Manager, 8-12 list

#### untranslatable, 8-20

#### using a nondefault termination character, 8-16

#### using in an input file, 8-15

#### entering

#### comments, 8-15

#### return characters, 8-15

#### including braces in text, 8-15

#### INSERT recording function, 8-15

#### record boundaries, 8-15

#### using special string equivalents for recording functions, 8-15

### STOP command, CD-164 to CD-165

- Stopping collection execution, 4-6, 5-25
- Storing DECwindows benchmark files
  - restriction, 3-27
- Storing DECwindows result files
  - restriction, 3-27
- Storing file
  - in CMS libraries, 7-3 to 7-4
  - outside the library, 7-2
- Subgroup, Glossary-8
- SUBMIT command, CD-166 to CD-168
- Submitting collections, 4-5
- Subprocess
  - attaching to, 6-24
  - creating, 6-24
- Successful comparison status, 4-10 table
- Supported DEC/Test Manager environments, 1-3
- Synchronization record, 9-6, 9-7
- SYNCHRONIZE command (DCL), 3-24
- Synchronize play back
  - DECwindows input file, 9-6
- Syntax
  - commands, CD-3 to CD-6
- SYS\$SHARE:DTMSHR.EXE shareable image, 10-8

## T

---

- Template directory
  - See also template file
  - canceling the default, 7-3, CD-132
  - default, 7-2
  - displaying the default, CD-150
  - establishing the default, CD-132
  - overriding the default, 7-3, CD-29, CD-132
  - setting, 3-4
- Template file, 1-4, 3-9, 7-2
  - See also Template directory
  - associating a file specification with, CD-91
  - default file name, CD-37
  - replacing, CD-91
  - setting template directories, 7-2 to 7-3
  - storing
    - in CMS libraries, 7-3 to 7-4
    - outside the library, 7-2
  - using DTM\$TEST\_NAME, 6-15
- Terminal characteristics, 3-25, 8-19
- Terminal characteristics block, 8-2, 8-7
- Terminal type, 3-25
  - in terminal characteristics block, 8-7
- Termination character, 3-17
  - default, CD-102
  - in Input files, CD-70
- Termination character (Cont.)
  - redefining, 3-19 to 3-20
  - using when extracting an input file, 8-16
- Test, 3-1, 3-7
  - comparing interactive, CD-12
  - copying, CD-20
  - create DECwindows, 2-5
  - creating noninteractive, 3-14 to 3-15
  - DECwindows, 3-7
    - replaying interactive tests, 3-23
    - restriction on playing, 3-25
  - description
    - See Test description
  - displaying
    - a benchmark file, CD-64
  - epilogue file, 6-2 to 6-4
  - executing, 4-1
  - expression, Glossary-9
  - group expression, 4-2
  - grouping, 6-6
  - initializing the environment, 3-21
  - interactive, 3-7
    - setting the screen size, 3-25
    - terminal characteristics, 3-25
    - terminal type, 3-25
  - noninteractive, 3-7
  - organizing into collections, 4-1 to 4-3
  - organizing into groups, 6-7 to 6-8
  - prologue file, 6-2 to 6-4
  - recording, 3-17 to 3-19
    - DECwindows tests, 2-6 to 2-9
  - removing from groups, 6-10
  - replaying interactive tests, 3-22 to 3-23
  - restrictions when creating interactive tests, 3-23
  - results
    - comparing, 4-9 to 4-10
    - displaying, 5-16 to 5-19
    - examining, 5-6
    - locating, 5-8 to 5-9
  - reviewing results, 5-1
  - running, 4-1
  - supported screen sizes, 3-20
  - terminal
    - replaying interactive tests, 3-23
  - types, 3-7
  - working with results, 5-21 to 5-24
- Test description, 3-8 to 3-13
  - adding
    - a filter, CD-88
    - an epilogue file to, CD-88
    - a prologue file to, CD-90

## Test description

### adding (Cont.)

- a variable to, CD-91
- canceling a prologue file, CD-90
- changing a variable value, CD-91
- copying, 3-10 to 3-11
- creating, 3-8 to 3-9, CD-37
- default benchmark file name, CD-38
- default template file name, CD-37
- deleting, 3-13, CD-59, CD-60
- displaying, 3-10, CD-152
- fields, 3-8 to 3-9 table
  - benchmark, 3-9
  - commands, 3-9
  - comparison type, 3-9
  - filters, 3-9
  - groups, 3-9
  - remark, 3-9
  - template, 3-9
  - test epilogue, 3-8
  - test name, 3-8
  - test prologue, 3-8
  - test type, 3-9
  - variables, 3-9

### inserting into a group, CD-81

### modifying, 3-12 to 3-13, CD-86

### noninteractive tests, 3-15

### removing

- a benchmark file specification from, CD-87
- an epilogue file from, CD-88
- a prologue file from, CD-90
- a remark from, CD-90
- a variable from, CD-91
- filters, CD-90

### removing from a group, CD-112

### replacing

- a benchmark file specification, CD-87
- a filter, CD-88
- an epilogue file, CD-88
- a prologue file, CD-90
- a remark, CD-90
- a template file specification, CD-91

### Test epilogue files, 6-3 to 6-4

### Test group expression, 4-2

### Test prologue files, 6-3

### Test system

- components, 3-1

### Text output request string, 9-6

### Timing interval

- recorded by DEC/Test Manager, 8-9

### Title bar, 2-1

## Translations performed

- translation tables, 8-21 to 8-35

### Type-ahead, 3-24, Glossary-10

## U

---

### UIC, 7-5

### Unsuccessful comparison status, 4-10 table

### UPDATE command, CD-210

### Updated comparison status, 4-10 table

### Updating a benchmark file, 5-21

- in DECwindows, 2-14

- in the Review subsystem, 5-21 to 5-22

### User identification code

- See UIC

## V

---

### Variable, 6-11 to 6-18

- adding to a test description, CD-91

### changing

- default value of, CD-95

- scope of, CD-94

- type of, CD-95

- usage of, CD-95

### changing the value of, CD-91

### creating, 6-11, CD-44

### defined by DEC/Test Manager, 6-15 to 6-18

### defining, CD-44

### deleting, 6-13, CD-63

### displaying summary information, CD-156

### DTM\$COLLECTION\_NAME, 6-15

### DTM\$DECW\$DISPLAY, 6-17

### DTM\$DELAY\_TIMEOUT, 6-17 to 6-18

### DTM\$OMIT\_PRINTABLE\_SCREEN, 6-18

### DTM\$RESULT, 6-16 to 6-17

### DTM\$TEST\_NAME, 6-15 to 6-16

### expression, Glossary-10

### global, 6-11 to 6-12, Glossary-4

### local, 6-12, Glossary-6

### modifying, 6-11, CD-94

### modifying characteristics, 6-13

### naming restrictions, 6-11

### overriding, 6-13 to 6-14

### removing from a test description, CD-63, CD-91

### removing the remark from, CD-95

### replacing the remark, CD-95

### VAX Code Management System

- See CMS

### Verify

- correcting library errors, CD-170



VERIFY command, CD-169 to CD-171  
Verifying library structure, 7-1 to 7-2  
Viewing benchmark images, 2-3  
Viewing differences, 2-3  
Viewing result images, 2-3  
View menu, 2-3  
Views, 2-2

expanding, 2-3  
selecting, 2-3  
using, 2-2 to 2-3

## **W**

---

Writing callback routines, 10-6



# How to Order Additional Documentation

---

## Technical Support

If you need help deciding which documentation best meets your needs, call 800-343-4040 before placing your electronic, telephone, or direct mail order.

## Electronic Orders

To place an order at the Electronic Store, dial 800-DEC-DEMO (800-332-3366) using a 1200- or 2400-baud modem. If you need assistance using the Electronic Store, call 800-DIGITAL (800-344-4825).

## Telephone and Direct Mail Orders

<b>Your Location</b>	<b>Call</b>	<b>Contact</b>
Continental USA, Alaska, or Hawaii	800-DIGITAL	Digital Equipment Corporation P.O. Box CS2008 Nashua, New Hampshire 03061
Puerto Rico	809-754-7575	Local Digital subsidiary
Canada	800-267-6215	Digital Equipment of Canada Attn: DECdirect Operations KAO2/2 P.O. Box 13000 100 Herzberg Road Kanata, Ontario, Canada K2K 2A6
International	_____	Local Digital subsidiary or approved distributor
Internal <sup>1</sup>	_____	USASSB Order Processing - WMO/E15 <i>or</i> U.S. Area Software Supply Business Digital Equipment Corporation Westminster, Massachusetts 01473

---

<sup>1</sup>For internal orders, you must submit an Internal Software Order Form (EN-01740-07).



# Reader's Comments

Guide to VAX DEC/Test Manager  
AA-Z330E-TE

Please use this postage-paid form to comment on this manual. If you require a written reply to a software problem and are eligible to receive one under Software Performance Report (SPR) service, submit your comments on an SPR form.

Thank you for your assistance.

<b>I rate this manual's:</b>	<b>Excellent</b>	<b>Good</b>	<b>Fair</b>	<b>Poor</b>
Accuracy (software works as manual says)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Completeness (enough information)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Clarity (easy to understand)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Organization (structure of subject matter)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Figures (useful)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Examples (useful)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Index (ability to find topic)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Page layout (easy to find information)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

I would like to see more/less \_\_\_\_\_

What I like best about this manual is \_\_\_\_\_

What I like least about this manual is \_\_\_\_\_

I found the following errors in this manual:

Page	Description
_____	_____
_____	_____
_____	_____

Additional comments or suggestions to improve this manual:

\_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_

I am using **Version** \_\_\_\_\_ of the software this manual describes.  
Name/Title \_\_\_\_\_ Dept. \_\_\_\_\_

Company \_\_\_\_\_ Date \_\_\_\_\_

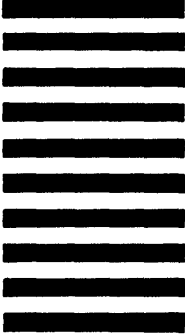
Mailing Address \_\_\_\_\_  
Phone \_\_\_\_\_

-- Do Not Tear - Fold Here and Tape

**digital**™



No Postage  
Necessary  
if Mailed  
in the  
United States



**BUSINESS REPLY MAIL**  
FIRST CLASS PERMIT NO. 33 MAYNARD MASS.

POSTAGE WILL BE PAID BY ADDRESSEE

DIGITAL EQUIPMENT CORPORATION  
Corporate User Publications—Spit Brook  
ZK01-3/J35  
110 SPIT BROOK ROAD  
NASHUA, NH 03062-9987



-- Do Not Tear - Fold Here